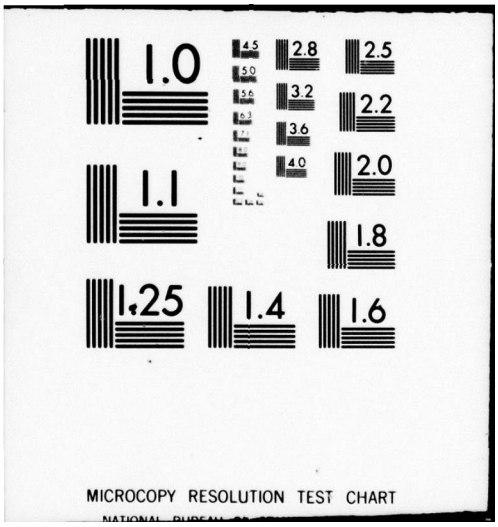
AD-A072 871 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON DC F/G 9/2
THE CCTC QUICK-REACTING GENERAL WAR GAMING SYSTEM (QUICK). PROG--ETC(U)
JUL 79

UNCLASSIFIED CCTC-CSM-MM-9-77-VI-CHG-2 NL

1 OF 2
ADA
072871


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY RESOLUTION TEST CHART

DEFENSE COMMUNICATIONS AGENCY
COMMAND AND CONTROL
TECHNICAL CENTER
WASHINGTON, D. C. 20301

3

IN REPLY
REFER TO: C314

LEVEL II

11 July 1979

27 June 79

TO: RECIPIENTS

SUBJECT: Change 2 to Program Maintenance Manual, CSM MM 9-77, Volume I, Data Management Subsystem

Pt 1 - A054377
Pt 2 - A054310

1. Insert the enclosed change pages and destroy the replaced pages according to applicable security regulations.

2. A list of Effective Pages to verify the accuracy of this manual is enclosed. This list should be inserted before the title page.

3. When this change has been posted, make an entry in the Record of Changes.

FOR THE DIRECTOR:

153 Enclosures
Change 2 pages

J. DOUGLAS POTTER
Assistant to the Director
for Administration

AD A072871

DDC FILE COPY

79 08 9 00

DEFENSE COMMUNICATIONS AGENCY
COMMAND AND CONTROL
TECHNICAL CENTER
WASHINGTON, D. C. 20301

IN REPLY
REFER TO: C124

(11) 27 July ~~1~~ 1979

MEMORANDUM FOR DISTRIBUTION

-A054377 A054310

SUBJECT: CSM MM 9-77, Volume I, Parts I and II, Change 2, Quick Reacting General War Gaming System, Data Management Subsystem

1. Enclosed are pages 901 and 901.1 (one sheet) of the referenced change 2. They were originally printed incorrectly.

2. Please pull and destroy pages 901 and 901.2 (one sheet) and insert the enclosed sheet.

E. Gilbert (12) 138 p.
for A.L. LAROCHE
Chief, Documentation and
Publication Control Branch

(6) The CCTC Quick-Reacting General War Gaming System, (QUICK). Program Maintenance Manual. Volume I. Data Management Subsystem. Change 2.

(14) CCTC-CSM-MM-9-77-V1-CHG-2

409 658

```
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/CNCLST
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/DATGRP
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/FLOCRS
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/MRVRST
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PRNPUT
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/RDMUL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/RDPRNZ
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/RDSET
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/RDSMAT
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/RNGALT
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/SETABLE
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/TIMEPRT
$       LINK    ALCMUL,ALCINT
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/ALCMUL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/MULCON
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/ADDSAL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/ASGOUT
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/BOMPRM
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/MYAPOS
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PRNTALL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PRNTCON
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PRNTNOW
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/TABLEMUP
$       LINK    FGD
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/FGD
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/FRSTGD
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/CRDCAL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/FLGCHK
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/INICRD
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/NXSPLT
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PKCALC
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PRNTOF
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/RECON
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/SETPAY
$       LINK    SGD,FGD
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/SGD
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/SCNDGD
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/NXSPLT
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/RECON
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/SETPAY
$       LINK    STAL,SGD
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/STAL
```

Figure 194. (Part 6 of 11)

```
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/STALL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/FORMATS
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/FMUP
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/LAMGET
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PREMIUMS
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PRNTOS
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/SALVAL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/SPLIT
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/WAD
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/WADOUT
$       LINK    DEFAL,STAL
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/DEFAL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/DEFALOC
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/FMUP
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/LAMGET
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PREMIUMS
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PRNTOD
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/RESVAL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/SALVAL
$       LINK    EVAL,ALOC
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/EVALALOC
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/EVALALOC
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/EVALPLAN
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/EVAL2
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/PREVAL
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/SSSPCALC
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/TGTMODIF
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/WPNMODIF
$       LINK    DGZSEL,EVAL
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/ALOCOUT
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/ALOCOUT
$       LINK    OFFSET
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/OFFSET
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/COMPRESS
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/CUMINV
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/DGZ
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/ERGOT1
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/FINDMIN
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/F2BMIN
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/GRADF
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/MOVE
```

Figure 194. (Part 7 of 11)

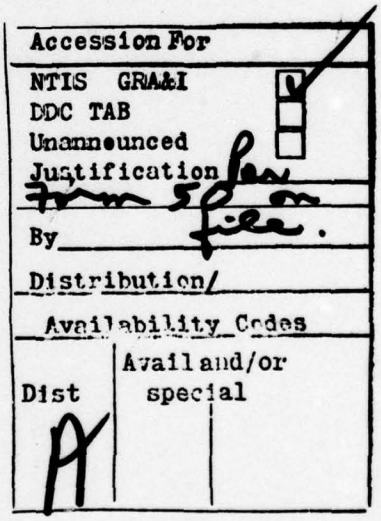
EFFECTIVE PAGES - MAY 1979

This list is used to verify the accuracy of CSM MM 9-77, Volume I after change 2 pages have been inserted. Original pages are indicated by the letter 0, change 1 pages by the numeral 1, and change 2 pages by the numeral 2.

Page No.	Change No.	Page No.	Change No.
Title Page, Part I	0	90.1-90.2	1
ii	2	91	2
iii-vii	1	92	0
viii	2	93	1
ix-x	0	94-99	0
xi	1	100-101	1
1-3	0	102	0
4	1	103-130	1
5-14	0	131-135	0
15-16	1	136-138	2
17	0	139-147	0
18	1	148-150	1
19-29	0	151-152	0
30	1	153-154	2
31	2	155-156	1
32-33	0	157-158	0
34	2	159-166	1
35	0	167-168	0
36	1	169-172	1
37-40	2	173	0
41	1	174	1
41.1-41.2	1	175	0
42-45	0	176	1
46	1	177-181	0
47-49	2	182	2
50	1	183-254	0
51	2	255	2
52-53	1	256-264	0
54	2	265	2
55	1	266	0
56	2	267	2
57-61	0	268-275	0
62	2	275.1-275.6	2
63-75	0	276-284	0
76	2	285	2
77-79	1	286-309	0
79.1-79.4	1	310	2
80-88	0	311-434	0
89-90	1	434.1-434.2	1

Page No.	Change No.	Page No.	Change No.
434.3-434.4	0	722-742	0
Title Page, Part II	0	743-744	1
ii	2	745-752	0
iii	1	753	2
iv-vi	2	754-763	0
vii	1	764	2
viii-ix	2	765-766	0
x	0	767-768	2
xi	1	769-780	0
435-438	0	780.1-780.2	2
439	1	781-782	0
440-444	0	783-785	2
445	1	786-791	0
446-546	0	792-793	2
547-548	1	794-800	0
549-558	0	801-802	2
559	2	803-810	0
560-572	0	811-815	2
572.1-572.6	1	816-839	0
573-580	0	840-841	2
581	1	842-847	0
582-589	2	848-852	2
590	0	853-855	0
591-593	1	856-857	2
593.1-593.2	1	857.1-857.6	2
594-595	1	858-871	0
596-632	0	872-873	2
633	2	874-891	0
634-665	0	892	2
666	2	893-895	0
667	0	896	2
668	2	897	2
669-675	0	898	2
676	2	899	1
676.1-676.20	1	900	2
676.21	2	901	1
676.22-676.38	1	901.1-901.2	2
677-678	0	901.3	2
679	2	901.4-901.5	2
680	0	901.6	1
681-682	2	902	2
683-694	0	903	1
695-699	2	904	2
700-712	0	905-910	1
713-714	2	910.1-910.2	1
715-718	0	911	1
719	1	911.1-911.2	1
720	0	912-919	1
721	1	919.1-919.6	1

Page No.	Change No.
920-921	0
921.1-921.6	1
922	0
923-924	1
925-926	0



ACKNOWLEDGMENT

This documentation was prepared under the direction of the Chief for Military Studies and Analysis, CCTC, in response to a requirement of the Studies, Analysis, and Gaming Agency, Organization of the Joint Chiefs of Staff. Technical support was provided by System Sciences, Incorporated under Contract Number DCA100-75-C-0019. Change set two was prepared under Contract Number DCA100-78-C-0035.

ILLUSTRATIONS (PART I)

Figure		Page
1	Major Subsystems of the QUICK System.....	3
2	Procedure and Information Flow in QUICK/HIS 6000....	4
3	Integrated Data Line Format.....	10
4	Example of Hierarchically Structured Data.....	12
5	Record Linkage Through Chains.....	12
6	Scenario Data Structure.....	37
7	Target Data Structure.....	38
8	Weapon Data Structure	39
9	Geographic Data Structure.....	41
9.1	Sortie Data Structure.....	41.1
10	Data Organization Index.....	42
11	Assignment Table.....	43
12	Miscellaneous Organizational Data.....	45
13	Program COP.....	63
14	Subroutine Banner.....	66
15	Subroutine ERPROC.....	69
16	Subroutine HDFND.....	71
17	Subroutine INICOP.....	77
17.1	Subroutine INPRIN.....	79.3
18	Subroutine INSPUT.....	82
19	Subroutine MODGET.....	90
20	Subroutine QDATA.....	92
21	Subroutine BOOT.....	105
22	Subroutine DCTFND.....	132
23	Subroutine MNMFND.....	135
24	Subroutine NUMFND.....	137
25	Subroutine RNMFND.....	140
26	Subroutine SEEKER.....	142
27	Subroutine STRMAK.....	144
28	Subroutine ERRFND.....	146
29	Subroutine LNGSTR.....	149
30	Subroutine SYNTAX.....	155
31	Subroutine TABINS.....	179
32	Subroutine WEBSTR.....	186
33	Subroutine INPTRN.....	190
34	Subroutine DELTAB.....	228
35	Subroutine INMATH.....	231
36	Subroutine LINEIO.....	237
37	Subroutine PARLEV.....	242
38	Subroutine TABGET.....	244
39	Subroutine CREAAT.....	252
40	Subroutine CHANGE.....	255
41	Subroutine ENTMOD (DATA).....	259
42	Subroutine VALPUT.....	261
43	Subroutine CHANGE: Step One.....	266
43.1	Subroutine CHANGE: Step One and a Half.....	275.1
44	Subroutine CHANGE: Step Two	276
45	Subroutine CHANGE: Step Three.....	286
46	Subroutine CHANGE: Step Four.....	288

Table 8. Example of Instruction Codes (Part 1 of 5)

Command

```
CHANGE WHERE SIDE=BLUE AND RANGE BETWEEN RANGE OF TYPE 'B-52' AND
     RANGE OF TYPE 'B-52' TIMES 10 SETTING CEP LIKE TYPE 'B-52'
     (SPEED,SPDLO) = (10000,9000)
```

ARRAY NUMBER	ARRAY VALUE	DESCRIPTION
1	4	Verb number - CHANGE
2	2	Number of Adverbs
3	22	Number of first Adverb - WHERE
4	7	Array number of beginning of WHERE clause
5	17	Number of second Adverb - SETTING
6	64	Array number of beginning of SETTING clause
		Beginning of WHERE clause, first group of instructions calculates RANGE OF TYPE 'B-52' TIMES 10
7	50*	Load numeric value
8	6	Numeric is an attribute
9	148	Address of attribute (RANGE)
10	125	Number of attribute (RANGE)
11	59	Attribute is modified by OF phrase - this is address of identifier attribute (TYPE)
12	51	Number of identifier attribute (TYPE)
13	9	Indicates value for identifying attribute is alphabetic
14	B-52	First half of value
15		Second half of value (= blanks)
16	66*	Multiply by numeric
17	10	Indicates numeric is a constant number
18	10.	Numeric constant
19	55*	Store calculated value in an internal variable
20	1	Index number of internal variable

*Instruction code.

Table 8. (Part 2 of 5)

ARRAY NUMBER	ARRAY VALUE	DESCRIPTION
		The next group of instruction evaluates the phrase SIDE=BLUE
21	49*	Load alphabetic value
22	6	Alphabetic is an attribute
23	2	Attribute's address (SIDE)
24	2	Attribute's number (SIDE)
25	0	Indicates no OF phrase
26	17*	Equal to alphabetic?
27	9	Alphabetic is alpha constant
28	BLUE	First half of alpha constant
29		Second half of alpha constant (= blanks)
30	42*	Store result of comparison in a logical variable
31	1	Logical variable index number
		The next group of instructions evaluates RANGE BETWEEN RANGE OF TYPE 'B-52' AND RANGE OF TYPE 'B-52' given that the final portion has already been calculated
32	50*	Load Numeric Value
33	6	Numeric is an attribute
34	148	Attribute's address (RANGE)
35	125	Attribute's number (RANGE)
36	0	No OF phrase
37	30*	Greater than or equal to numeric?
38	6	Numeric is an attribute
39	148	Attribute's address (RANGE)
40	125	Attribute's number (RANGE)
41	59	Address of identifier in OF phrase (TYPE)

*Instruction code.

Table 8. (Part 3 of 5)

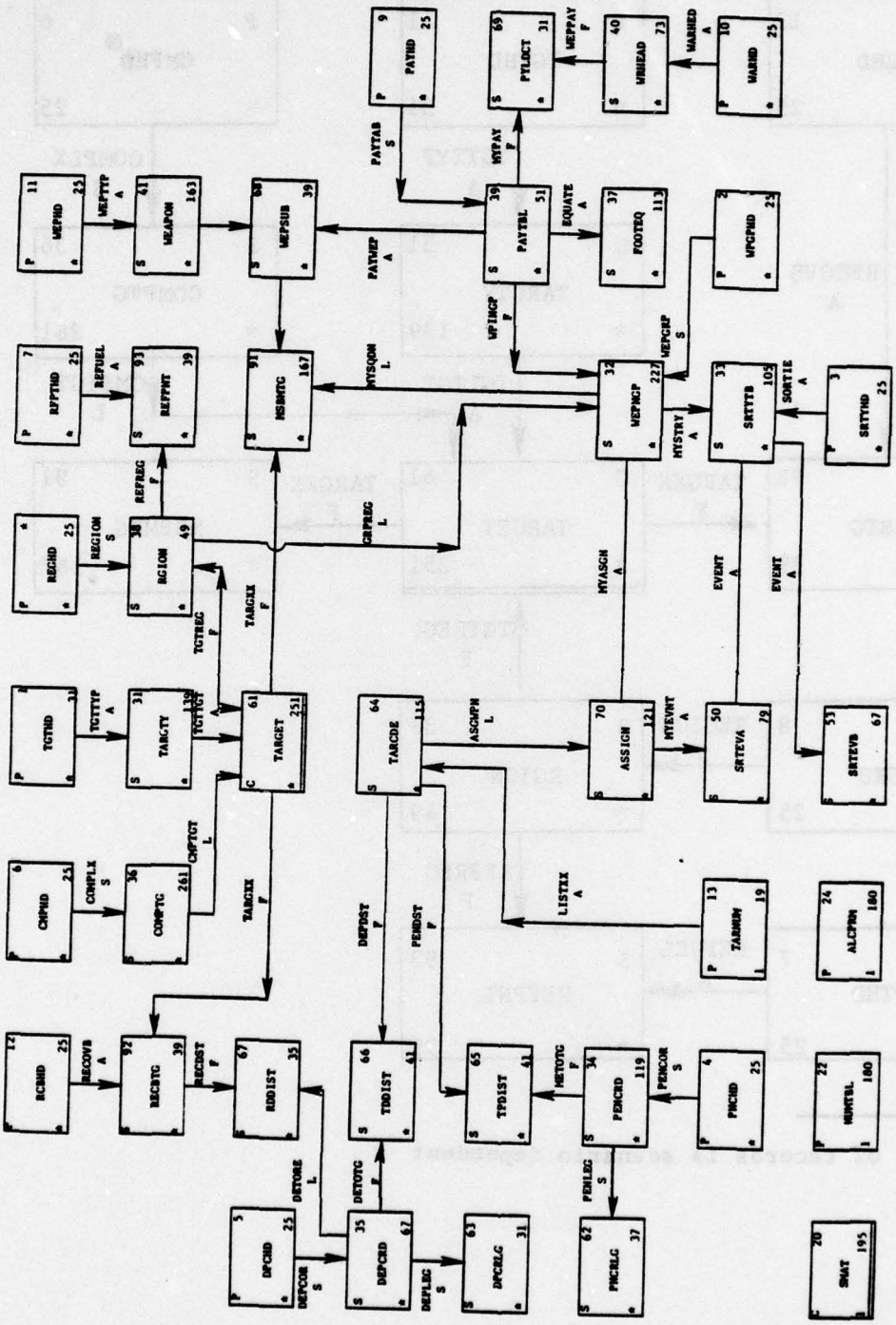
ARRAY NUMBER	ARRAY VALUE	DESCRIPTION
42	51	Number of identifier (TYPE)
43	9	Value for identifier is alphabetic
44	B-52	First half of alpha value
45		Second half of alpha value (= blanks)
46	43*	Logical AND indicates that result of previous comparison is to be logically 'anded' to following comparison. Thus word 47=0
47	0	
48	50*	Load Numeric Value
49	6	Numeric is an attribute
50	148	Attribute's address (RANGE)
51	125	Attribute's number (RANGE)
52	0	No OF phrase
53	39*	Less than or equal to an internal variable?
54	1	Index number of internal variable
55	42*	Store the results of the logical 'and' of the two comparisons in a logical variable
56	2	Index number of logical variable
		Now the previously set logical variables are used to evaluate the WHERE clause
57	41*	Load value of logical variable
58	1	Index number of logical variable
59	43*	AND to value of logical variable
60	2	Index number of logical variable
61	42*	Store result in logical variable
62	3	Index number of logical variable
63	1*	End of clause

*Instruction code.

Table 8. (Part 4 of 5)

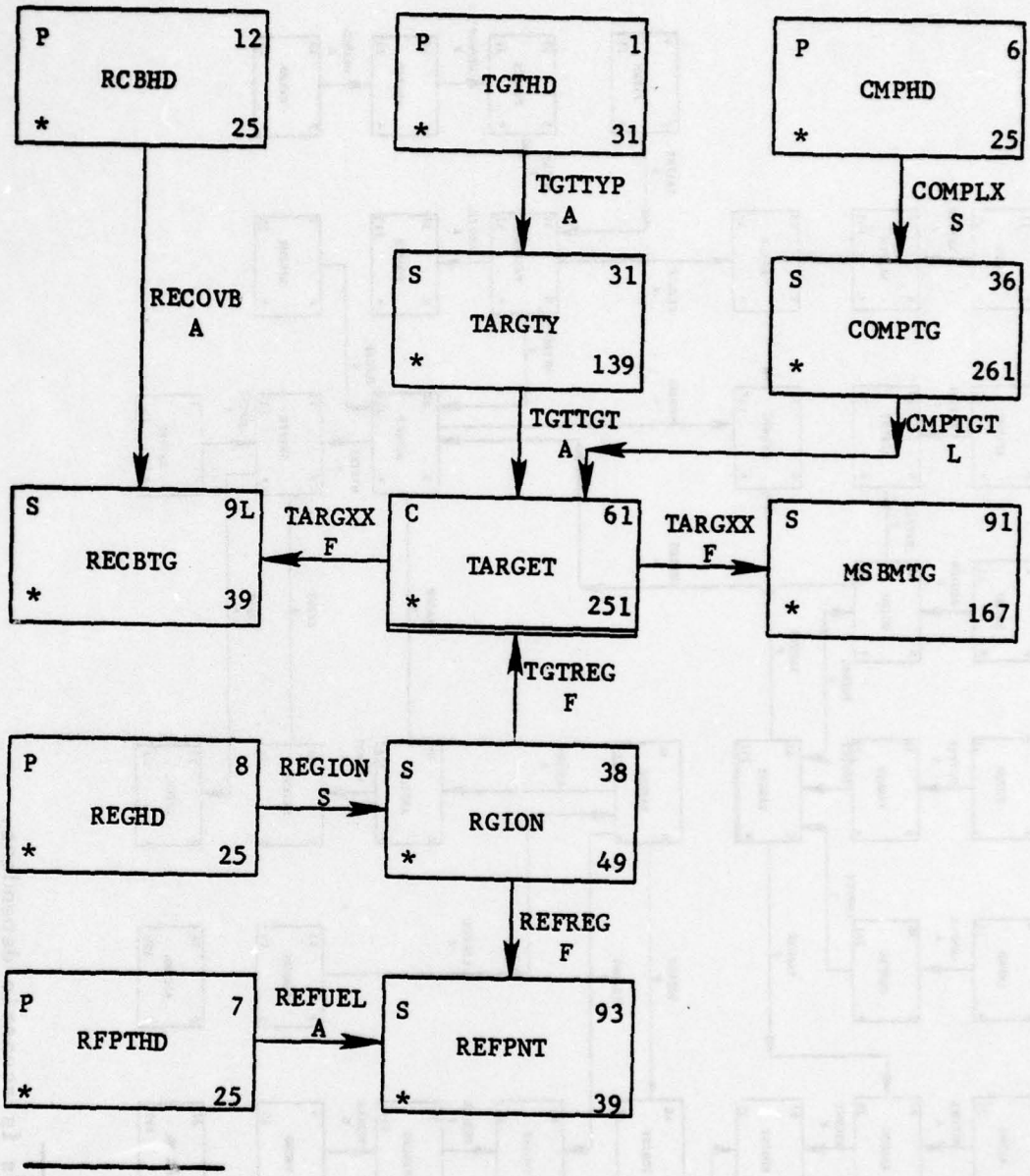
ARRAY NUMBER	ARRAY VALUE	DESCRIPTION
		Beginning of SETTING clause. First group of instructions is for CEP LIKE TYPE TYPE 'B-52'
64	50*	Load Numeric
65	6	Numeric is an attribute
66	145	Attribute's address (CEP)
67	122	Attribute's number (CEP)
68	0	No OF phrase
69	11*	LIKE string follows - Indicates that loaded attribute should be set to the value of the same attribute in identified record
70	59	Identifier attribute's address (TYPE)
71	51	Identifier attribute's number (TYPE)
72	9	Identifier's value is alphabetic
73	9	Identifier's value is alphabetic
74	B-52	First half of alphabetic value
75		Second half of alphabetic value (= blanks)
76	2*	End of Phrase
		The Last group of instruction is for (SPEED,SPDLO) = (10000,90000)
77	50*	Load Numeric
78	6	Numeric is an attribute
79	146	Attribute's address (SPEED)
80	123	Attribute's number (SPEED)
81	0	No OF phrase
82	18*	Set equal to numeric
83	10	Numeric is a constant
84	1000.	Numeric constant
85	43*	Logical AND used as phrase connector

*Instruction code.



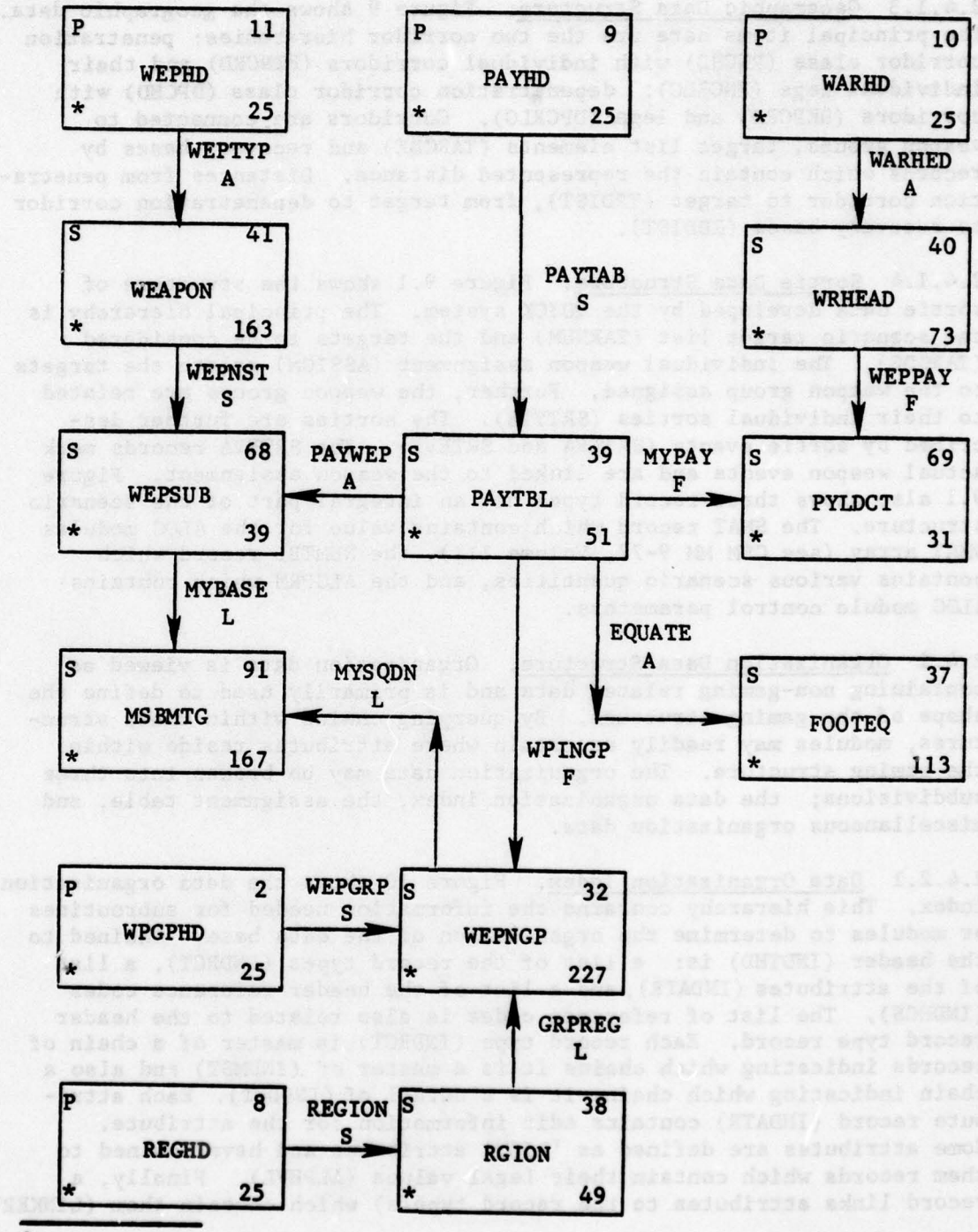
*Number of records is scenario dependent.

Figure 6. Scenario Data Structure



*Number of records is scenario dependent

Figure 7. Target Data Structure



*Number of records in scenario dependant

Figure 8. Weapon Data Structure

2.4.1.3 Geographic Data Structure. Figure 9 shows the geographic data. The principal items here are the two corridor hierarchies: penetration corridor class (PNCHD) with individual corridors (PENCRD) and their individual legs (PNCRLG); depenetration corridor class (DPCHD) with corridors (DEPCRD) and legs (DPCRLG). Corridors are connected to weapon groups, target list elements (TARCDE) and recovery bases by records which contain the represented distance. Distances from penetration corridor to target (TPDIST), from target to depenetration corridor to recovery bases (RDDIST).

2.4.1.4 Sortie Data Structure. Figure 9.1 shows the structure of sortie data developed by the QUICK system. The principal hierarchy is the scenario target list (TARNUM) and the targets to be considered (TARCDE). The individual weapon assignment (ASSIGN) relate the targets to the weapon group assigned. Further, the weapon groups are related to their individual sorties (SRTYTB). The sorties are further described by sortie events (SRTEVA and SRTEVB). The SRTEVA records mark actual weapon events and are linked to the weapon assignment. Figure 9.1 also shows three record types not an integral part of the scenario structure. The SMAT record which contains value for the ALOC modules SMAT array (see CSM MM 9-77, Volume III), the NUMTBL record which contains various scenario quantities, and the ALCPRM which contains ALOC module control parameters.

2.4.2 Organization Data Structure. Organization data is viewed as containing non-gaming related data and is primarily used to define the shape of the gaming structure. By querying chains within these structures, modules may readily ascertain where attributes reside within the gaming structure. The organization data may be broken into three subdivisions; the data organization index, the assignment table, and miscellaneous organization data.

2.4.2.1 Data Organization Index. Figure 10 shows the data organization index. This hierarchy contains the information needed for subroutines or modules to determine the organization of the data base. Chained to the header (INDTHD) is: a list of the record types (INDRCT), a list of the attributes (INDATR), and a list of the header reference codes (INDHCS). The list of reference codes is also related to the header record type record. Each record type (INDRCT) is master of a chain of records indicating which chains it is a master of (INDMST) and also a chain indicating which chains it is a detail of (INDDET). Each attribute record (INDATR) contains edit information for the attribute. Some attributes are defined as 'LIST' attributes and have chained to them records which contain their legal values (ALPHVL). Finally, a record links attributes to the record type(s) which contain them (LINKER).

2.4.2.2 Assignment Table. Figure 11 shows the Assignment Table. This hierarchy contains the information used to determine several attributes for targets based on information from a JAD format record. Chained to the header (ASNTAB) for each side are records containing the valid

Table 9. (Part 2 of 2)

RECORD TYPE NUMBER	RECORD TYPE NAME	DESCRIPTION (ATTRIBUTES)	CLASS VALUES
14	INDTHD	Data Organization Index Header (CLASS) (This is a CALC record)	INDEX
15	ASNTAB	Assignment Table Header (CLASS,SIDE)	ASSIGN
16	DCTHD	Dictionary Header (CLASS)	DICTON
17	SYNHD	Syntax Directory Verb Header (CLASS)	SYNTAX
18	ADVHD	Syntax Directory Adverb Header (CLASS)	ADVERB
19	MODTAB	Module Link Table (CLASS, link Table (100 words))	MODTAB
20	SMAT	SMAT Array (CLASS) (This is a CALC record)	SMAT
21	DISPHD	REPORT Module Display Header (CLASS)	DISPLY
22	NUMTBL	General Number Table (CLASS)	NUMBER
23	TABLST	Utility Table Header (CLASS) (This is a CALC record)	TABLST
24	ALCPRM	ALOC Control Parameters (CLASS)	ALCPRM

Table 10. Data Base Record Types - Secondary Records (Part 1 of 4)

RECORD NUMBER	TYPE NAME	DESCRIPTION (ATTRIBUTES)
31	TARGTY	Target Type (CNTRYLO, CNTRYOW, FLAG, FVALT1, FVALT2, FVALT3, FVALT4, FVALT5, FVULN1, NHRDCOMP, NTIMCOMP, T1, T2, T3, T4, T5, TYPE, VULN1 VULN2)
32	WEPNGP	Weapon Group (ATTINC, EXPASM, GBASE, GFRASM, GLAT, GLONG, GNWPNADJ, GNWPNS, GNVEH, GPKNAV, GREFCODE, GREFTIME, GROUP, GSBL, GSBLREAL, GSTART, GTYPE, GTYPREFC, GYIELD, IALERT, IREG, MAXSAL, NFIXES, NSAL, NSFIX, NUMALOC) (GROUP is used as a Match Key)
33	SRTYTB	Sortie Table (SDELAY, SDEPEN, SINDEXNO, SLAT, SLONG, SLOW, SLOW1, SLOW2, SLOW3, SORTNO, SREFUEL, SVEHNUM)
34	PENCRD	Penetration Corridor (ATTRCO, ATTRPRE1, ATTRPRE2, ATTRPRE3, ATTRSU, CORNUM, DEFDIST1, DEFDIST2, DEFDIST3, DEFRAN, HILOAT, KORSTY, NPRCRDEF, ORLAT, ORLONG)
35	DEPCRD	Depenetration Corridor (CORNUM, MYRECOV1, MYRECOV2, MYRECOV3, MYRECOV4)
36	COMPTG	Complex Target (CNTRYLO, CNTRYOW, DESIG, FLAG, FVALT1, FVALT2, FVALT3, FVALT4, FVALT5, FVULN1, HAZ2, HGZ, HGZ2, ICOMPL, IDHOB, INDEXNO, LAT, LONG, MAXFRA, MAXKILL, MINKILL, MISDEF, NAME, NHRDCOMP, NTIMCOMP, NTINT, RADIUS, TARDEFHI, TARDEFLO, TASK, TGTMULT, TGTNUMB, T1, T2, T3, T4, T5, VALUE, VOZ) (ICOMPL is a match-key)
37	FOOTEQ	Footprint Equation (Contains one hundred words which are module defined)
38	RGION	Region (IREG, CCREL) (IREG is a match key)
39	PAYTBL	Payload table (PAYTBLNM)
40	WRHEAD	Warhead (CEPASM, CPASMZRO, FFRAC, NAREADEC, NCMS, NDECOYS, NWHDS, PAYALT, PDUD, RANGEASM, RELASM, SPEEDASM, TYPE, YIELD)

Table 10. (Part 2 of 4)

RECORD NUMBER	TYPE NAME	DESCRIPTION (ATTRIBUTES)
41	WEAPON	Weapon type (ACTIVE, ALTDLY, BALC, CEP, CEPMIN, CMISS, FUNCTI, IPENMO, IRECMO, IREP, LCHINT, MAXSAL, NALTDLY, NMPSIT, PDES, PFPF, PINC, PLABT, RRABT, RANGE, RANGEDEC, RANGEREF, REANG, REL, RNGMIN, SIMLUN, SPDLO, SPEED, TOFMIN, TTOS, TYPE) (TYPE is a match key)
42	INDRCT	Index Record Type Record (contains record type name and number)
43	INDATR	Index Attribute Record (ATTRIBN1, ATTRIBN2, ATTRIBNO, ATTRBTYP, ATTRIBAD, ATDEFALT, ATTRNGHI, ATTRNGLO) (ATTRIBN1 and ATTRIBN2 are match keys)
44	INDMST	Index Master Record (CHAINNAM, MASDETNM, MASDETNO)
45	DCTTAB	Dictionary Tab-character (TABCHAR) (TABCHAR is a match key)
46	SYNVRB	Syntax Verb (CLAUSESW, VERBVAL) (VERBVAL is a matcky key)
47	PRMADV	Syntax Adverb (ADVERBVL, CLAUSETY, PHRASETY) (ADVERBVL is a match key)
48	TABLEZ	Utility Table (contains 100 words which are modu defined)
49	INDDET	Index Detail Record (CHAINNAM, MASDETNM, MASDETNO)
50	SRTEVA	Sortie Event Type A (LAT, LONG, SCUMSURV, SCHANGE, SDAMEXP, ADELTIME, SEVCODE, SLOCATTR, SPLACE)
51	ASNCLS	Assignment Table Class (ACLASS)
52	DISPRC	Display Record (IDISPNAM1, DISPNAM2)
53	SRTEVB	Sortie Event Type B (Same attributes as record number 50)
61	TARGET	Target (BENO, CATCODE, DESIG, HAZ, HAZ2, HGZ, HGZ2, ICOMPL, IDHOB, IGIW, INDEXNO, IREG, ISITE, LAT, LONG, MAJOR, MAXFRA, MAXKILL, MINKILL, MINOR, MISDEF, NAME, NTINT, POP, RADIUS, SIDE, TARDEFHI, TARDEFLO, TASK, TGTNUMB, VALUE, VOZ, WACNO) (Record is a CALC record - randomized on DESIG)

Table 10. (Part 3 of 4)

RECORD NUMBER	TYPE NAME	DESCRIPTION (ATTRIBUTES)
62	PNCRLG	Penetration Corridor Leg (ATTRLE, DOGLEG, LAT, LONG)
63	DPCRLG	Depenetration Corridor Leg (DOGLEG, LAT, LONG)
64	TARCDE	Target List Element (TGTNUMB, TGTREFCD)
65	TPDIST	Distance from Target to Penetration Corridor (ATTRCD, DISTANCE)
66	TDDIST	Distance from Target to Depenetration Corridor (DISTANCE, DISTDF)
67	RDDIST	Distance from Recovery Base to Depenetration Corridor (DISTANCE)
68	WEPSUB	Weapon Subtype (PAYTBLNM) (PAYTBLNM is a match key)
69	PYLDCT	Count of warhead type in Payload Table (NUMLOAD)
70	ASSIGN	Assignment of Weapon Group to Target (ARRIVE, ASGHOB, DGZLAT, DGZLONG, FIXED, FLMULT, FSALVO, GROUP, KORR, OFFLAT, OFFLONG, PEN, RVAL, TGTNUMB)
71	DCTWRD	Dictionary Word (WORDSTR1, WORDSTR2, WORDTY, WORDVL)
72	SYNCLZ	Links Verbs to Adverbs (ADVERBVL, VERBVAL)
73	ADVELM	Gives Legal Elements for elemental Adverbs (ELEMNTTY, ELEMNTVL)
74	LINKER	Links Attributes to Record Types (ATTRIBN1, ATTRIBN2, ATTRIBAD, ATTRBTYP, contains 2 words which are not attributes)
75	ALPHVL	Legal Values for 'LIST' Attributes (ALPLSTVL)
76	INDHCS	Stores Header reference codes (contains header reference code (two words), and the associated CLASS and SIZE values)
77	DISPDT	Display table list (contains 100 words which are created by REPORT)

Table 10. (Part 4 of 4)

RECORD NUMBER	TYPE NAME	DESCRIPTION (ATTRIBUTES)
78	ASNCTY	Assignment Table Country (COUNTRY, REGION)
79	ASNTYP	Assignment Table Type (ATYPE)
80	ASNDES	Assignment Table Desig (DESIGA2, KOUNT1, KOUNT2, KOUNT3, KOUNT4, KOUNT5)
91	MSBMTG	Missile Bomber Target (ADBLI, ADBLR, ALRTDB, ALRTDL, GROUP, IREFUEL, NADBLI, NADBLR, NLRTDB, NLRTDL, NOALER, NOINCO, NOPERSQ, NPRSQ1, NPRSQ2, NPRSQ3, NPRSQ4, NUMDBL, PAYTBLNM, PKNAV, VONBASE, WEPNAME)
92	RECBTG	Recovery base (CAPACITY)
93	REFPNT	Refuel Point (LAT, LONG, IREG)
94	ASNREC	Assignment Table Category (ASNTASK, CATHI, CATLO, CNFLG, MINCAP
95	TYPDES	Assignment Table Type DESIG (DESIGA2, FULL1, FULL2, FULL3, FULL4, FULL5)

Table 11. Data Base Chains (Part 1 of 4)

CHAIN NAME	MASTER RECORD	DETAIL RECORD	DESCRIPTION
ADVADV	ADVHD	PRMADV	Links Adverb Header to adverbs
ADVERB	PRMADV	SYNCLZ	Links adverb to link to verb
AILINK	INDATR	LINKER	Links Attribute Record to link to record type
ALCLAS	ASNTAB	ASNCLS	Links Assignment table header to assigned classes
ALLDES	ASNTAB	ASNDES	Links Assignment table header to assigned DESIG
ALTDES	ASNTYP	TYPDES	Links Assigned type to link to DESIGs for that type
ALTYPE	ASNCLS	ASNTYP	Links Assigned class to assigned types in that class
ASGWPN	TARCDE	ASSIGN	Links target to fix assignments to it
ASNRNG	ASNCTY	ASNREC	Links Assigned country to category ranges for that country
ATRIB	INDTHD	INDATR	Links index header to attribute record
CLAUSE	SYNVRB	SYNCLZ	Links verb to link to adverbs
CMPTGT	COMPTG	TARGET	Links complex to targets which make up the complex
COMPLX	CMPHD	COMPTG	Links complex header to complexes
CONTRY	ASNTAB	ASNCTY	Links assignment table header to assigned countries
DEPCOR	DPCHD	DEPCRD	Links depenetration corridor header to depenetration corridors
DEPDST	TARCDE	TDDIST	Links target to distance to depenetration corridors
DEPLEG	DEPCRD	DPCRLG	Links depenetration corridors to its doglegs
DESTYP	ASNDES	TYPDES	Links assigned DESIG to link to assigned type
DETORE	DEPCRD	RDDIST	Links depenetration corridor to distance to recovery base

CH-1

Table 11. (Part 2 of 4)

CHAIN NAME	MASTER RECORD	DETAIL RECORD	DESCRIPTION
DETOTG	DEPCRD	TDDIST	Links depenetration corridor to distance to target
DSPITM	DISPRC	DISPDT	Links display table to its elements
DSPLAY	DISPHD	DISPRC	Links display header to display tables
ELEMNT	PRMADV	ADVELM	Links elemental adverb to its legal elements
EQUATE	PAYTBL	FOOTEQ	Links footprint equations to the proper payload table
EVENT	SRTYTB	SRTEVA	Links sortie table to event type A (weapon event)
EVENT	SRTYTB	SRTEVB	Links sortie table to event type B (non-weapon event)
GRPREG	RGION	WEPNGP	Links region to weapon groups in that region
IALINK	INDRCT	LINKER	Links record type to link to attributes
IRDET	INDRCT	INDDET	Links record type to record showing chains of which it is a detail
IRMAST	INDRCT	INDMST	Links record type to record showing chains of which it is master
LISTXX	TARNUM	TARCDE	Links target list header to elements of the list
METOTG	PENCRD	TPDIST	Links penetration corridor to distance to target
MYASGN	WEPNGP	ASSIGN	Links weapon group to fixed assignments
MYBASE	WEPSUB	MSBMTG	Links weapon sub-type to missile/bomber targets that are its bases
MYEVNT	ASSIGN	SRTEVA	Links weapon assignments to their sortie event
MYNAMZ	INDRCT	INDHCS	Links record type for headers to their reference codes
MYPAY	PAYTBL	PYLDCT	Links payload table to warhead type count
MYSRTY	WEPNGP	SRTYTB	Links weapon group to its sortie tables

Table 11. (Part 3 of 4)

CHAIN NAME	MASTER RECORD	DETAIL RECORD	DESCRIPTION
MYSQDN	WEPNGP	MSBMTG	Links weapon group to missile/bomber targets which provide bases for the group
NAMEZ	INDTHD	INDHCS	Links index header to header reference codes
PAYTAB	PAYHD	PAYTBL	Links payload table header to payload tables
PAYWEP	PAYTBL	WEPSUB	Links payload table to weapon sub-type that uses it
PENCOR	PNCHD	PENCRD	Links penetration corridor header to penetration corridors
PENDST	TARCDE	TPDIST	Links target to distance to penetration corridor
PENLEG	PENCRD	PNCRLG	Links penetration corridor to its doglegs
RCTYP	INDTHD	INDRCT	Links index header to record types
RECDST	RECBTG	RDDIST	Links recovery base to distance to depen-tration corridors
RECOVB	RCBHD	RECBTG	Links recovery base header to recovery bases
REFREG	RGION	REFPNT	Links region to refuel points in the region
REFUEL	RFPTHD	REFPNT	Links refuel point header to refuel points
REGION	REGHD	RGION	Links region header to regions
SORTIE	SRTYHD	SRTYTB	Links sortie header to sortie tables
TAB	DCTHD	DCTTAB	Links dictionary header to its tab char-acters
TABXYZ	TABLST	TABLEZ	Links utility table header to utility tables
TARGXX	TARGET	MSBMTG	Links target to missile/bomber target additional data
TARGXX	TARGET	RECBTG	Links target to recovery base additional data
TGTREG	RGION	TARGET	Links region to targets in region
TGTTGT	TARGTY	TARGET	Links target type to targets of that type

Table 11. (Part 4 of 4)

CHAIN NAME	MASTER RECORD	DETAIL RECORD	DESCRIPTION
TGTTYP	TGTHD	TARGTY	Links target header to target types
TYPRNG	ASNTYP	ASNREC	Links assigned type to category range for that type
VALIST	INDATR	ALPHVL	Links 'list' attribute to its legal values
VERB	SYNHD	SYNVRB	Links syntax header to verbs
WARHED	WARHD	WRHEAD	Links warhead header to warhead types
WEPGRP	WPGPHD	WEPNGP	Links weapon group header to weapon groups
WEPNST	WEAPON	WEPSUB	Links weapon type to weapon subtype
WEPPAY	WRHEAD	PYLDCT	Links warhead type to count in payload table
WEPTYP	WEPHD	WEAPON	Links weapon header to weapon types
WORD	DCTTAB	DCTWRD	Links tab character to words with that tab
WPINGP	PAYTBL	WEPNGP	Links payload table to weapon groups using that table

Table 12. Chains Which are Linked to Master

CHAIN NAME	MASTER RECORD	DETAIL RECORD
ADVERB	PRMADV	SYNCLZ
AILINK	INDATR	LINKER
ALTDES	ASNTYP	TYPDES
ASGWPN	TARCDE	ASSIGN
ASNRNG	ASNCTY	ASNREC
CMPTGT	COMPTG	TARGET
DEPDST	TARCDE	TDDIST
DETORE	DEPCRD	RDDIST
DETOTG	DEPCRD	TDDIST
DESTYP	ASNDES	TYPDES
IALINK	INDRCT	LINKER
METOGP	PENCRD	GPDIST
METOTG	PENCRD	TPDIST
MYASGN	WEPNGP	ASSIGN
MYBASE	WEPSUB	MSBMTG
MYEVNT	ASSIGN	SRTEVA
MYNAMZ	INDRCT	INDHCS
MYSQDN	WEPNGP	MSBMTG
MYSRTY	WEPNGP	SRTYTB
PAYWEP	PAYTBL	WEPSUB
PENDST	TARCDE	TPDIST
RECDST	RECBTG	RDDIST
TARGXX	TARGET	MSBMTG
TARGXX	TARGET	RECBTG
TGTREG	RGION	TARGET
TGTTYP	TARGTY	TARGET
TGTTGT	TARGTY	TARGET
WARHED	WARHD	WRHEAD
WEPPAY	WRHEAD	PYLDCT
WPINGP	PAYTBL	WEPNGP

Table 14. Internal COP Common Block

BLOCK	ARRAY OR VARIABLE	DESCRIPTION
C25		Represents record type INDHCS. Each record is used to identify the BCD reference code of a header.
	XCLASS	Header's CLASS value
	XSIDE	Header's SIDE value
	XREFCD	Header's BCD reference code (this variable is type character *8)
C35	LINKS(100)	Module Link Table
FIRST	INCOM	Logical switch which, when on, indicates a sentence is being analyzed
SYMBOL	KYMBOL	Used to pass constructed ERRFND symbol (see section 3.5.2).
TABLZ		Contains buffers for utility tables used to store ERRFND table
	KTBVAL(100,4)	Buffer of 100 words for each table type
	TBRFCD(20,4)	Contains Reference Codes for tables
	NUMOT(4)	Number of utility tables created
	NUMCT(4)	Index numbers of table currently in buffer.

3.7 Main Routine of COP

PURPOSE: Main program of executive module

ENTRY POINTS: COP (for purpose of discussion)

FORMAL PARAMETERS: None

COMMON BLOCKS: C15, IPQT, OOPS

SUBROUTINES CALLED: CLZIDS, DELTAB, ERRFND, INICOP, INPTRN, INSDEL, MODGET

CALLED BY: HIS operating system

Method:

First, several switches are set: CHCKOV to control a later call to DELTAB, ERROR to indicate no error has occurred yet, and ENDSW to indicate no end of input. Next, INICOP is called. The process which follows occurs for each command sentence.

First the ERRFND overlays are read in (this does not occur for the first sentence since INICOP has already done this). Next, ERRFND is called. If CHCKOV is still true, the INPTRN link is read in and INPTRN is executed if no error has occurred. If an error occurred before INPTRN, only DELTAB is called.

Next, if no error has occurred, MODGET is called to execute the module. If an error has occurred, CHCKOV is set to false.

Finally, the End Input switch is checked (ENDSW); if not on, the next input sentence is processed. If it is on, CLZIDS is called and processing stops.

COP is illustrated in figure 13.

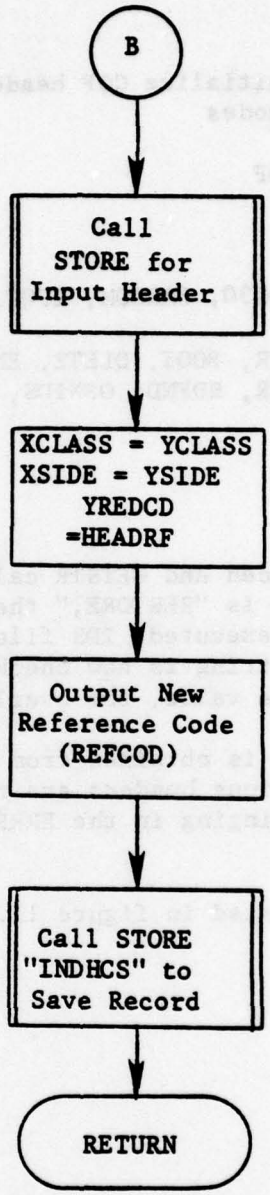


Figure 16. (Part 5 of 5)

3.7.4 Subroutine INICOP

PURPOSE: To initialize COP headers and check for special run modes

ENTRY POINTS: INICOP

FORMAL PARAMETERS: None

COMMON BLOCKS: C15, C30, ERRCOM, IPQT, OOPS, STRING

SUBROUTINES CALLED: BANNER, BOOT, DLETE, ENTMOD(SRM), ERPRIN, GETSTR, HDFND, OPNIDS, RETRV, STORE, WEBSTR

CALLED BY: COP

Method:

The standard header is produced and GETSTR called for the first string. If the first string of input is "RESTORE," the overlay for the Save and Restore Module (SRM) is executed, IDS file opened and GETSTR is called again. The current string is now checked for the value "INITIALIZE." If this is the value, the overlay for BOOT is executed.

In any case, the next string is obtained from GETSTR and the utility tables purged. Finally, various headers are retrieved and the syntax analysis process begun by bringing in the ERRFND overlay and calling WEBSTR.

Subroutine INICOP is illustrated in figure 17.

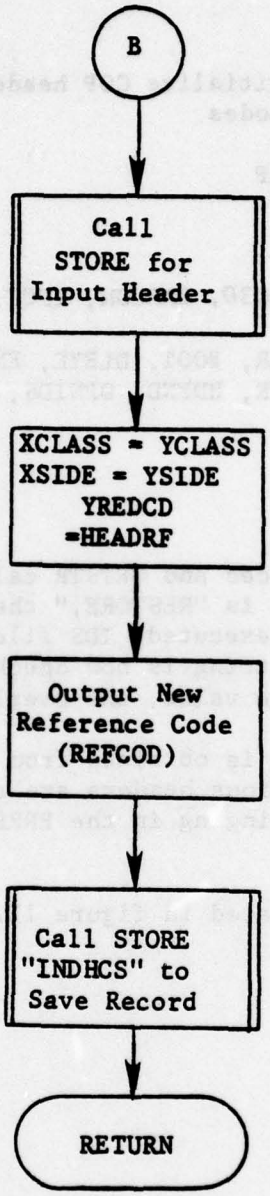


Figure 16. (Part 5 of 5)

3.7.4 Subroutine INICOP

PURPOSE: To initialize COP headers and check for special run modes

ENTRY POINTS: INICOP

FORMAL PARAMETERS: None

COMMON BLOCKS: C15, C30, ERRCOM, IPQT, OOPS, STRING

SUBROUTINES CALLED: BANNER, BOOT, DLETE, ENTMOD(SRM), ERPRIN, GETSTR, HDFND, OPNIDS, RETRV, STORE, WEBSTR

CALLED BY: COP

Method:

The standard header is produced and GETSTR called for the first string. If the first string of input is "RESTORE," the overlay for the Save and Restore Module (SRM) is executed, IDS file opened and GETSTR is called again. The current string is now checked for the value "INITIALIZE." If this is the value, the overlay for BOOT is executed.

In any case, the next string is obtained from GETSTR and the utility tables purged. Finally, various headers are retrieved and the syntax analysis process begun by bringing in the ERRFND overlay and calling WEBSTR.

Subroutine INICOP is illustrated in figure 17.

3.7.7 Subroutine QDATA

PURPOSE: Performs IDS functions

ENTRY POINTS: CLEANP, CLZIDS, DIRECT, DLETE, HEAD, MODFY, NEXTTT, OPNIDS, RETRV, STORE

FORMAL PARAMETERS: ARGUMENT: Record type or chain name

COMMON BLOCKS: C10, C15, C20, C25, C30, C35, C40, C50

SUBROUTINES CALLED: ERPROC

CALLED BY: (IDS utility entry, called by all QUICK modules)

Method:

The entry points for this routine fall into three groups: those with no argument: CLEANP, CLZIDS, OPNIDS, DIRECT; those whose argument is a record type name: DLETE, MODFY, RETRV, STORE; and those whose argument is a chain name: HEAD, NEXTTT.

CLEANP, CLZIDS, OPNIDS and DIRECT

These entries each carry out a single function. CLEANP dumps altered pages to file storage. CLZIDS closes the IDS file. OPNIDS opens the IDS file for update. DIRECT retrieves the record whose binary reference code is stored in C10.

DLETE, MODFY, RETRV and STORE

For each of these entry points the process is quite similar. The input record type name is looked up in a table of legal record type names (REC-TYPE-TABLE). If it is not in the table the error code is set to '000RRR'. After the look-up, the subroutine branches to the appropriate code to perform the requested function. In some cases (see notes to figure 20) the requested function is not allowed in which event the error code is set to '000ILC'.

HEAD and NEXTTT

For each of these entry points the process is similar. The input chain name is looked up in a table of legal chain names (CHAIN-NAME-TABLE). If it is not in the table, the error code is set to '000CCC'. After the look-up, the subroutine branches to the appropriate code to perform the requested function.

After any of the above entry points process have taken place, the error code is checked. If an error has occurred ERPROC is called.

Subroutine QDATA is illustrated in figure 20.

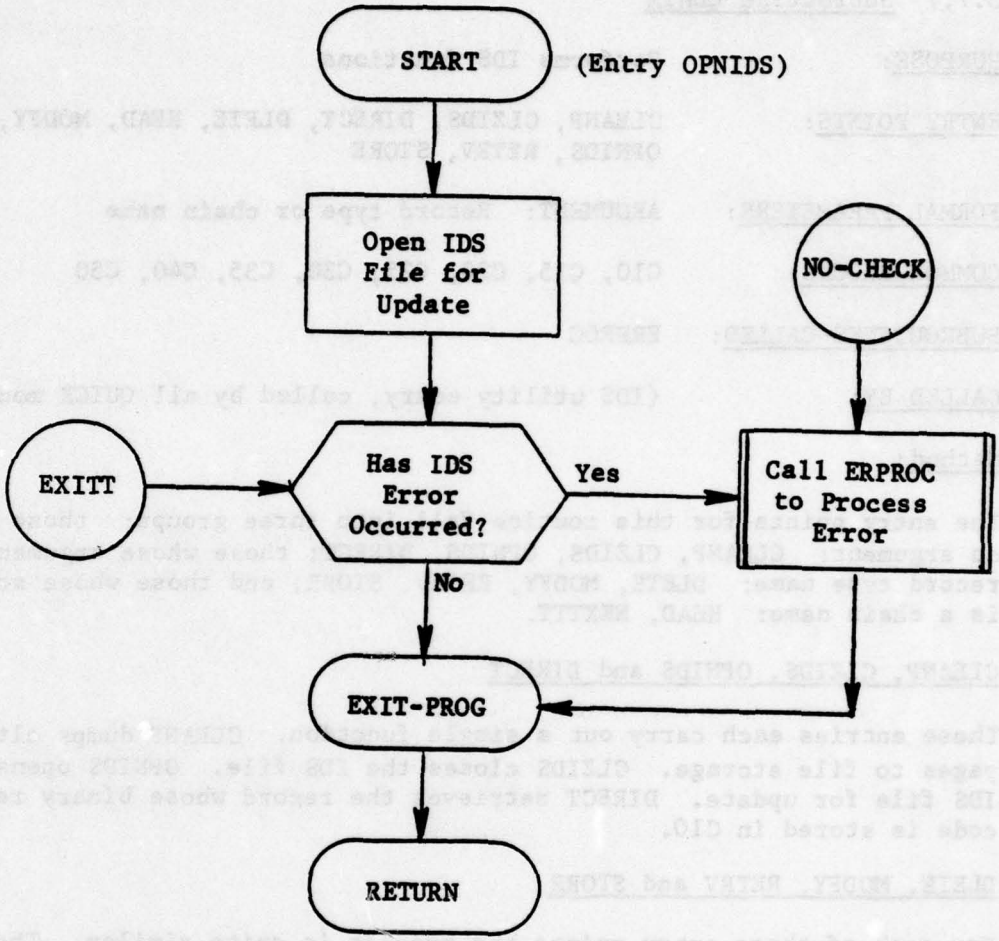


Figure 20. Subroutine QDATA: Entry OPNIDS (Part 1 of 9)

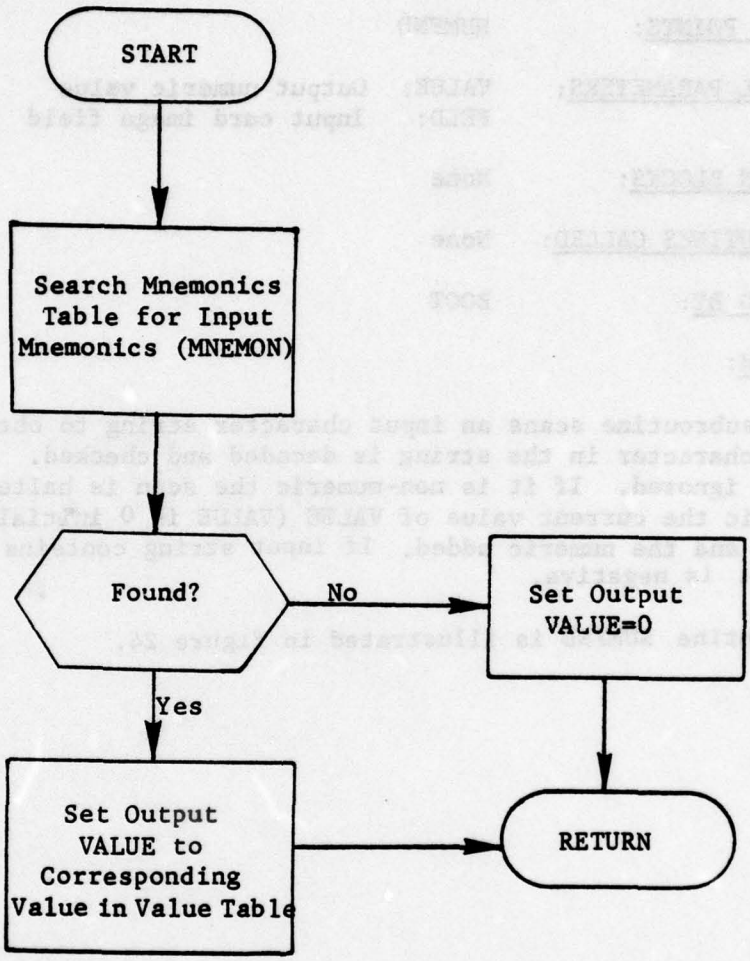


Figure 23. Subroutine MNMFND

3.8.3 Subroutine NUMFND

PURPOSE: Finds number in field of input card image for BOOT

ENTRY POINTS: NUMFND

FORMAL PARAMETERS: VALUE: Output numeric value
FELD: Input card image field

COMMON BLOCKS: None

SUBROUTINES CALLED: None

CALLED BY: BOOT

Method:

This subroutine scans an input character string to obtain a number. Each character in the string is decoded and checked. If it is a blank it is ignored. If it is non-numeric the scan is halted. If it is numeric the current value of VALUE (VALUE is 0 initially) is multiplied by 10 and the numeric added. If input string contains a minus sign, result is negative.

Subroutine NUMFND is illustrated in figure 24.

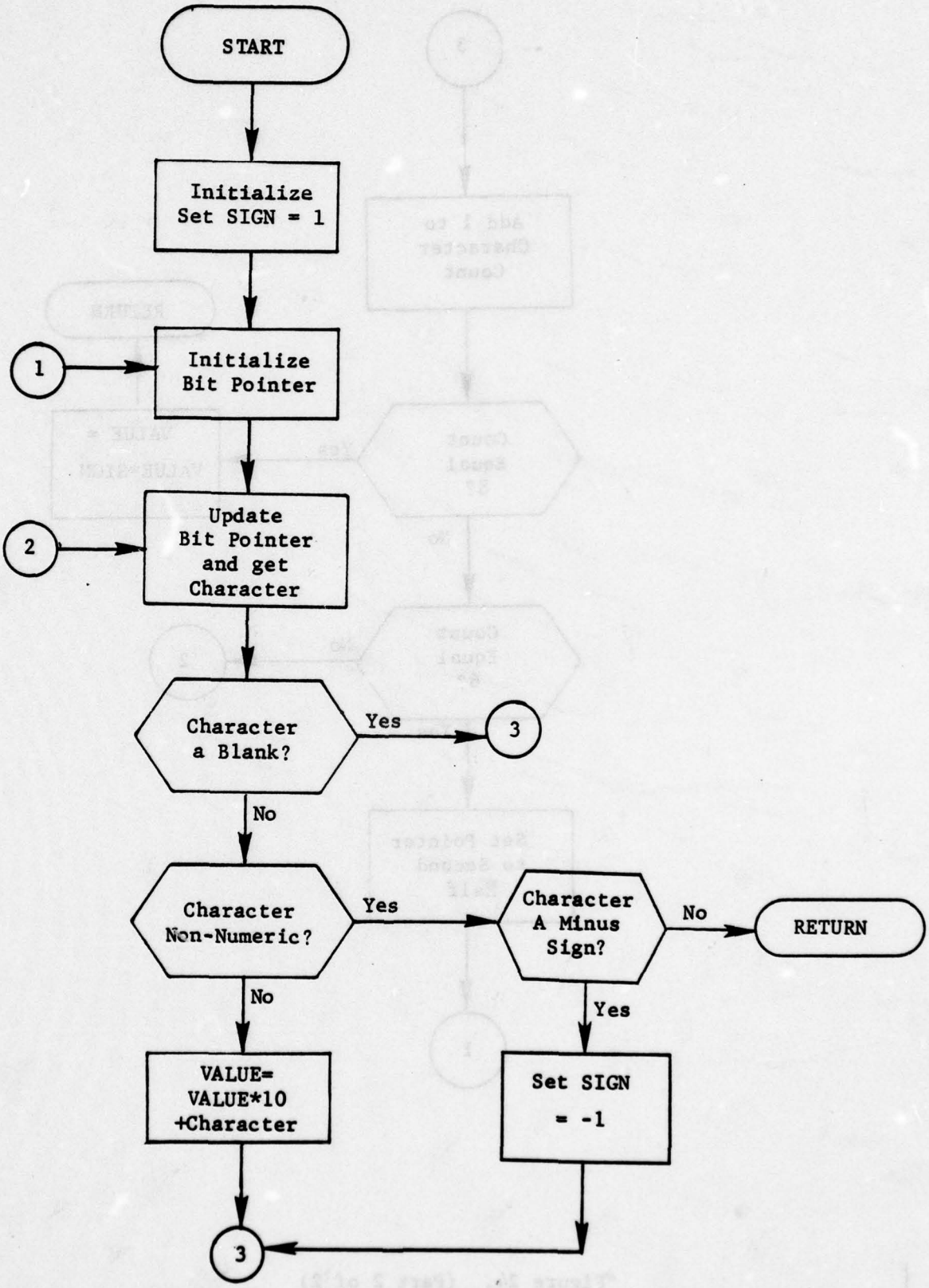


Figure 24. Subroutine NUMFND (Part 1 of 2)

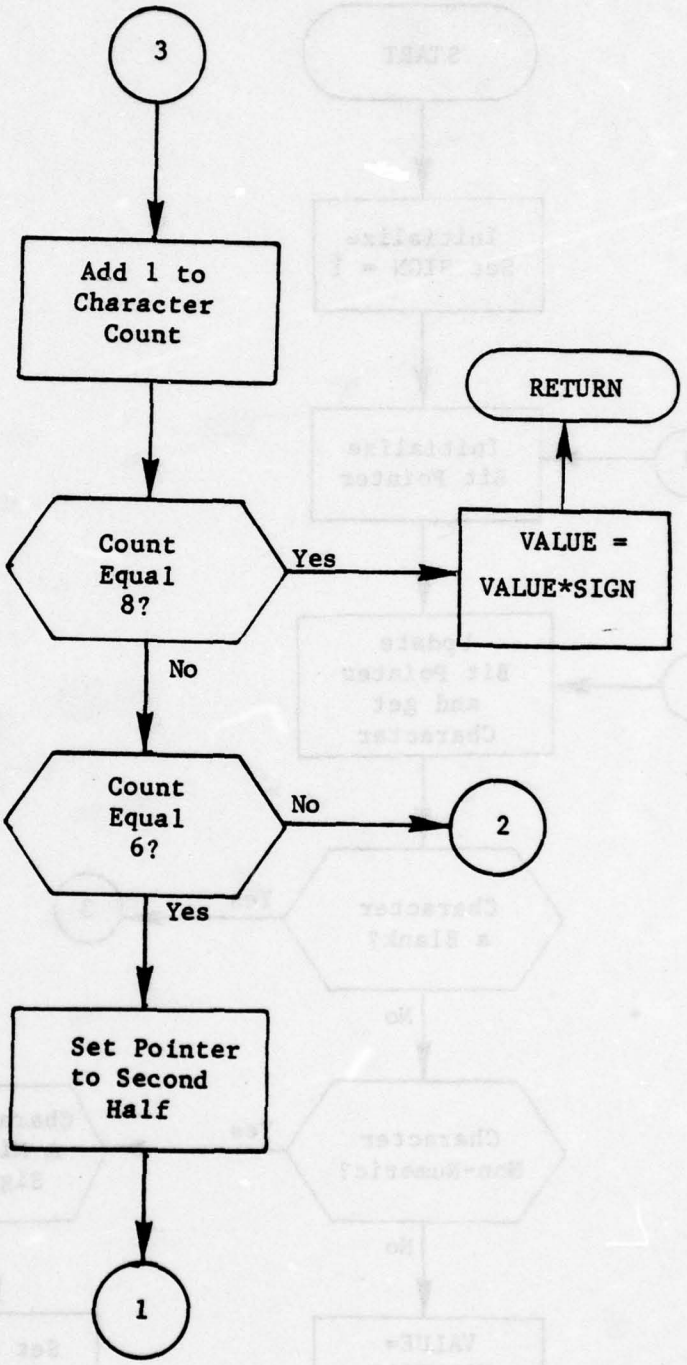


Figure 24. (Part 2 of 2)

3.9.2 Subroutine SYNTAX

PURPOSE: Analyze syntax of input command sentences

ENTRY POINTS: SYNTAX

FORMAL PARAMETERS: None

COMMON BLOCKS: C10, C15, C30, FIRST, IPQT, OOPS, STRING

SUBROUTINES CALLED: ERPRIN, HDFND, HEAD, NEXTTT, RETRV

CALLED BY: ERRFND

Method:

This subroutine checks each input string against the current setting of various switches. If the string is one the types of strings expected, the appropriate switches are reset. The following switches and counters are used by SYNTAX:

BETWEN - True when processing BETWEEN relation
BLPRCT - Boolean parenthesis level count
BOLTYP - True when expecting relational phrase, left parenthesis or NOT (boolean clause only)
BOOL - True when clause is boolean
CONTEQ - True when an equals relation is being continued in a boolean clause
ELEMNT - True when phrases are elemental
ELRT - 1 - if elemental non-boolean clause
2 - if elemental boolean clause
ENDCLZ - True if clause may end
ENDCOM - True if sentence may end
EQUALS - True when processing EQUALS relation
INCLZ - True when processing a clause
INCOM - True when processing a sentence
LIKE - True when processing a LIKE relation
PHPNT - Relational phrase branch
1 - expecting operation unless inside collection
2 - expecting value or collection
3 - check for equal and between continuation
PHPRCT - Relational phrase parenthesis count
RELPHR - True when processing a relational phrase
RESTRC - True when phrase type is restricted relational (EQUALS or LIKE)
SINGLE - True when clause type is single
SNGCT - Count of phrases for single type clause

VALCT - Value element branch
1 - expecting end of element or OF
2 - expecting identifying attribute
3 - expecting alphabetic or numeric constant
VALEL - True if processing value element
VALQE - True if processing value expression
VALTYP - True when expecting new value element or left parenthesis
VLPRCT - Value expression parenthesis count

Processing proceeds as follows:

First the string is checked to see if it is a verb, if not, branch to statement 5 (see figure 30). If so INCOM is checked. If INCOM is true but ENDCOM is false an error has occurred. Otherwise INCOM is set to false and the subroutine exists. If INCOM is false, the VERB chain is searched to find a match. When the match is found INCOM and ENDCOM are set to two and INCLZ is set to false. If the clause switch indicates that the verb must have a clause (ICSW=1), ENDCOM is set to false.

Statement 5 (figure 30)

If the string is not an adverb, a branch is made to statement 19. If it is an adverb INCOM is checked and if false an error has occurred. Next, if INCLZ is true but ENDCLZ is false an error has occurred. Next, the CLAUSE chain is searched to see if this adverb is matched to the current verb. If it is HEAD is called to relieve the adverb's record and the clause and phrase type (IXTYP and ILTYP) are packed into the adverb's value. Now various switches are set depending on the clause type (IXTYP). If boolean, set SINGLE to false, BOOL to true, BLPRCT to zero and BOLTYP to true. If sequence, set SINGLE and BOOL to false. If single, set SINGLE to true, BOOL to false, SNGCT to zero. If null, set INCLZ to false and ENDCLZ and ENDCOM to true. For all but the last, set INCLZ to true and ENDCLZ, ENDCOM and VALQE to false and branch on phrase type (ILTYP). For relational phrases set RESTRC to false, for restricted phrases set RESTRC to true, for both set RELPHR, ELEMNT, EQUALS, LIKE, BETWEN and CONTEQ to false, and PHPRCT to zero. For elemental phrases set ELEMNT to true.

Statement 19 (figure 30)

At this point refer to the flow chart, figure 30 as processing is best illustrated therein. However, some sections that are of particular note follow.

Statement 52 (figure 30)

Here many relational phrases have ended. The LIKE phrase has one more element--the value for the identifiers attribute. The BETWEEN phrase may have an optional 'AND'. After this or without it the BETWEN switch

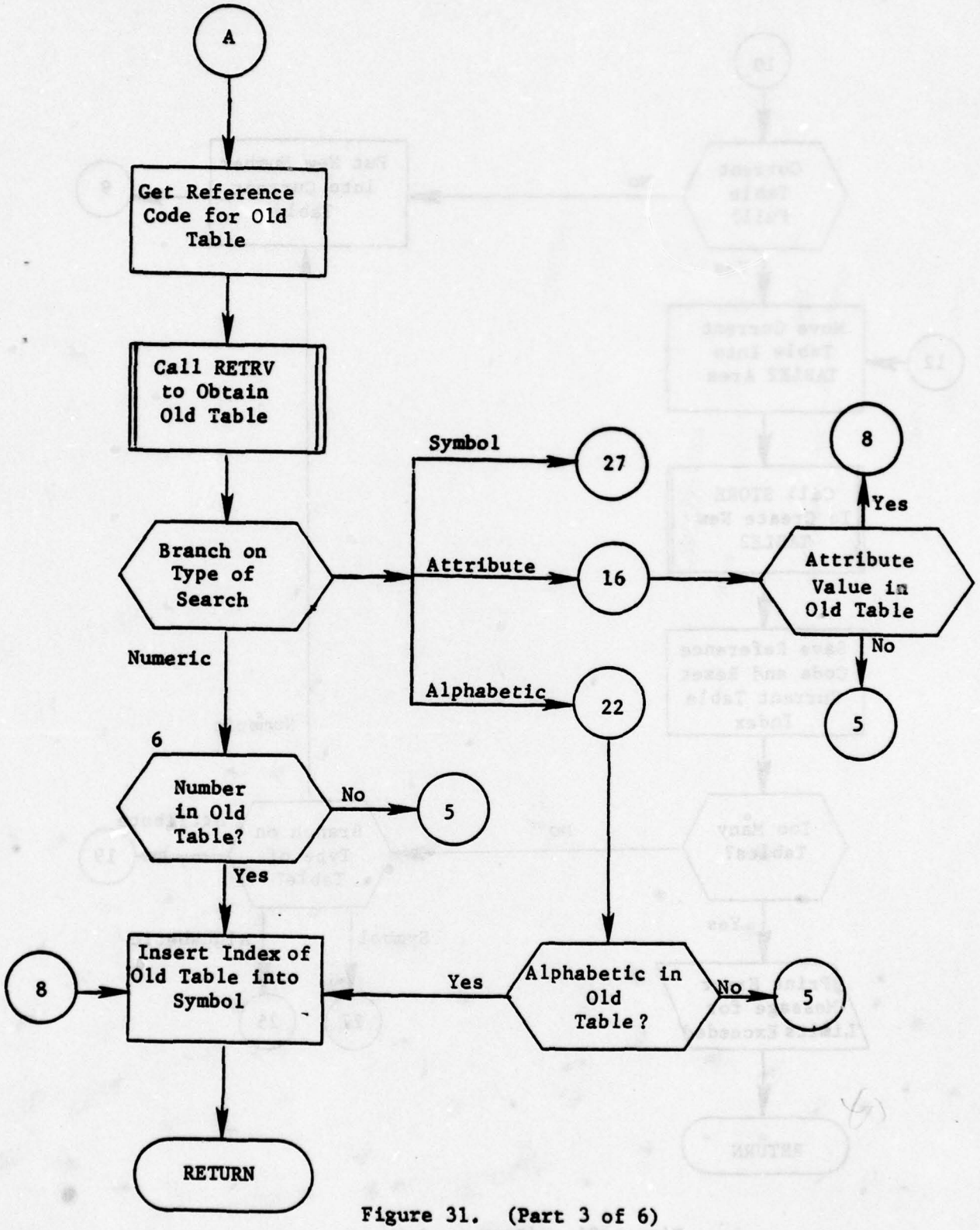


Figure 31. (Part 3 of 6)

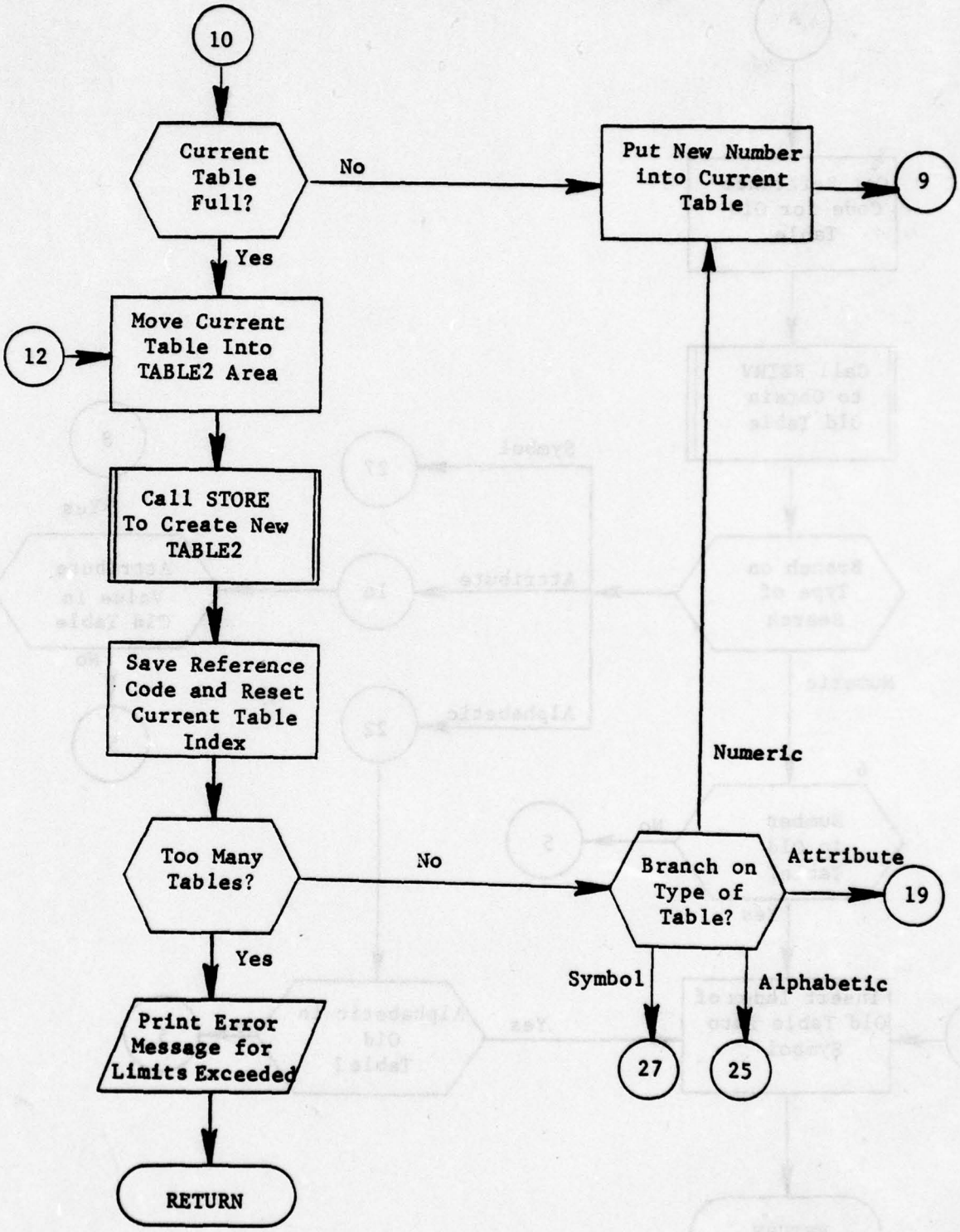


Figure 31. (Part 4 of 6)

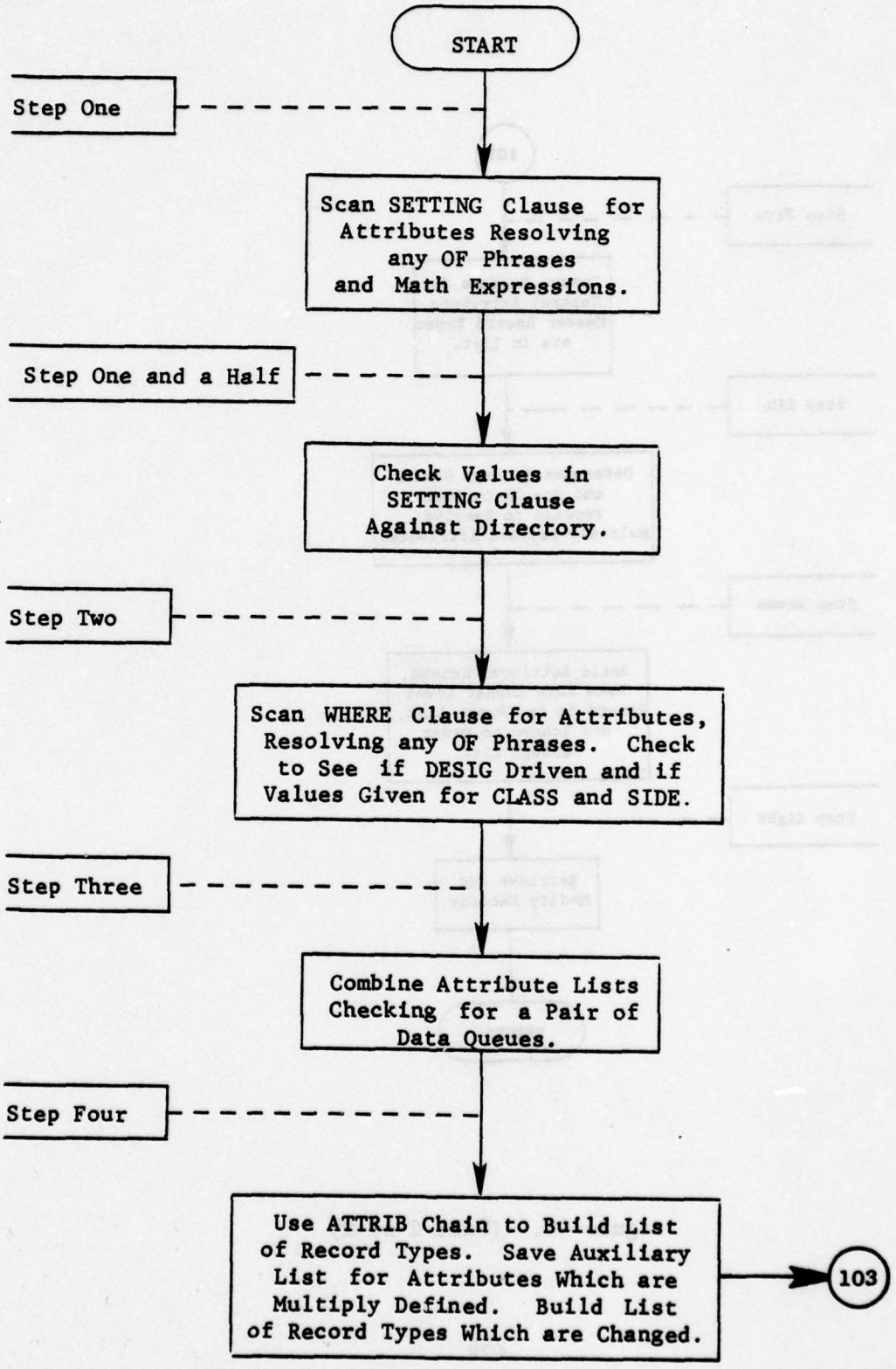


Figure 40. Subroutine CHANGE (Macroflowchart) (Part 1 of 2)

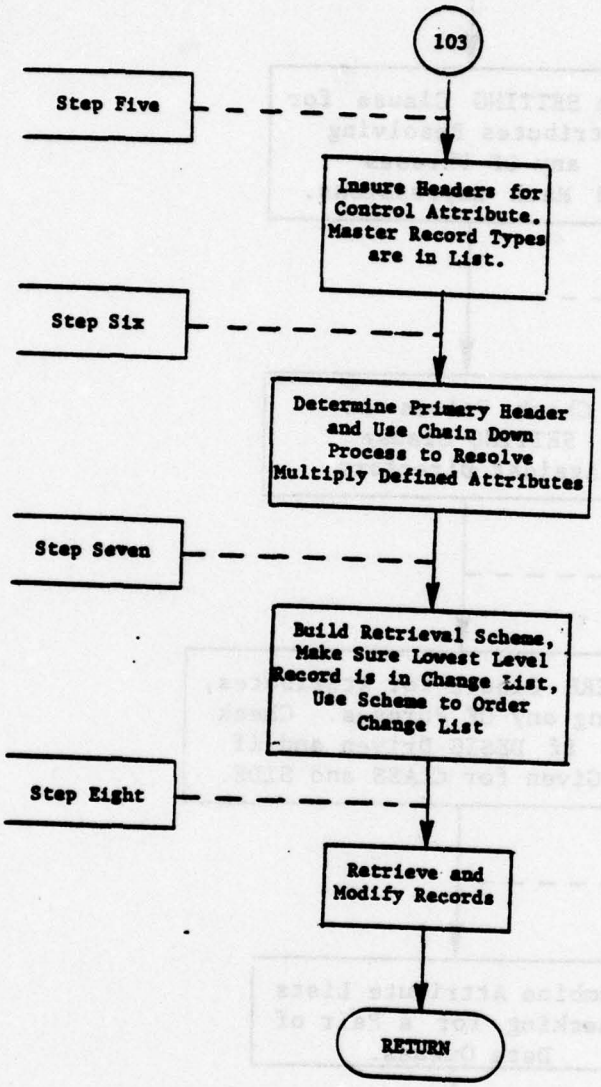


Figure 40. (Part 2 of 2)

4.8 Subroutine CHANGE*

PURPOSE: To change existing records

ENTRY POINTS: CHANGE

FORMAL PARAMETERS: None

COMMON BLOCKS: C10, C20, C30, ERRCOM, OOPS, ORDER, PRINSP, SCHEME

SUBROUTINES CALLED: DESSCH, GETNXT, HDFND, HEAD, INSGET, LINKUP, MODFY, NEXTTT, NXTDES, OFVAL, PRIMHD, SETSCH, UNCODE, VALDEL, VALFND, VALGET, VALPUT, XMATH, XWHERE

CALLED BY: ENTMOD (DATA)

Method:

The CHANGE verb process may be broken into eight steps which are carried out in sequence (see figure 40).

Step One

The SETTING clause is scanned. In the process any OF phrases and LIKE strings are immediately resolved using VALFND and their values stored via OFVAL. Also the extent of any mathematical calculations is noted. as the limits for execution for subroutine XMATH. The main thrust of the step is to list any attributes (ATNUMB) and save the values assigned to them. If an extended equals phrase is included, a queue of the values is built and the involved attribute(s) are flagged (ATTYPE=2) (see figure 43).

Step One and a Half

The SETTING clause is scanned. In the process, any attribute which is simply set to a value is looked up in the directory and the value checked. List alphabetics have their list of valid values compared to their input values. Numeric attributes have their values compared to the limits given in the directory. Any errors are reported but processing continues.

Step Two

The WHERE clause is scanned. In the process any OF phrases and LIKE strings are immediately resolved using VALFND and their values stored via OFVAL. Any attributes are saved in a list (WHATNB). If an extended equals phrase is included, its involved attributes are flagged (WHATYP=4) and its values stored in a queue via VALPUT. The CLASS and SIDE attributes are particularly noted and their values saved to assist in the retrieval scheme construction process. If the DESIG attribute is the only attribute included this fact is noted as the retrieval process differs greatly in this event (see figure 44).

*The main routine of overlay DATACH

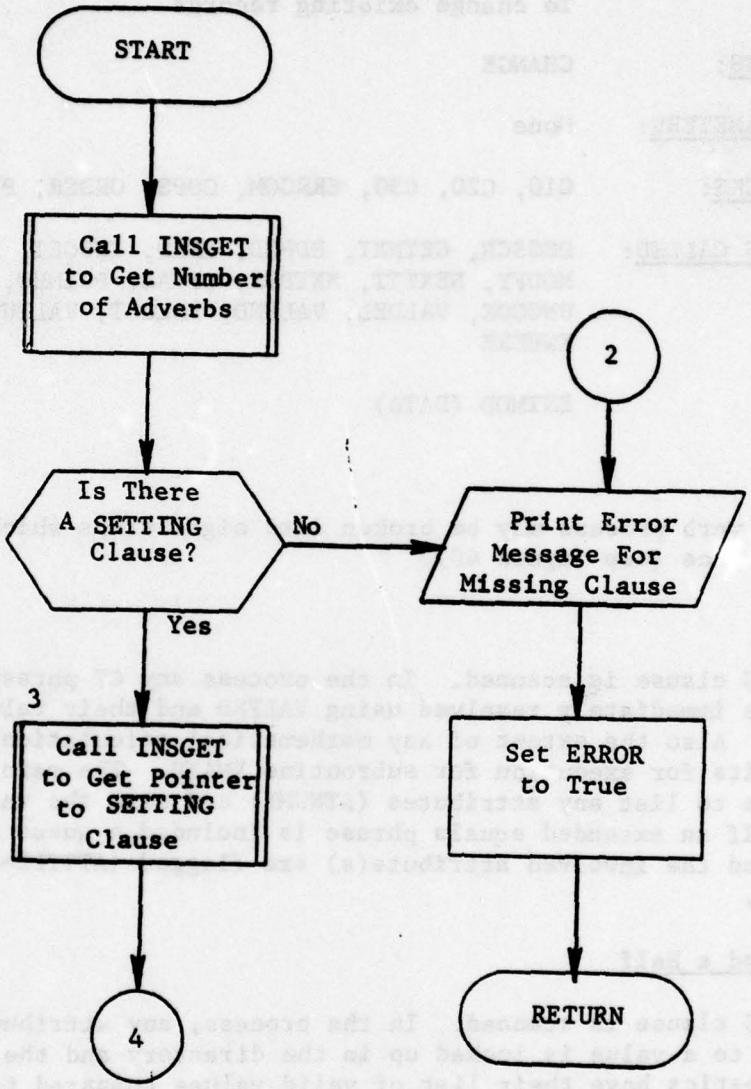


Figure 43. Subroutine CHANGE: Step One (Part 1 of 10)

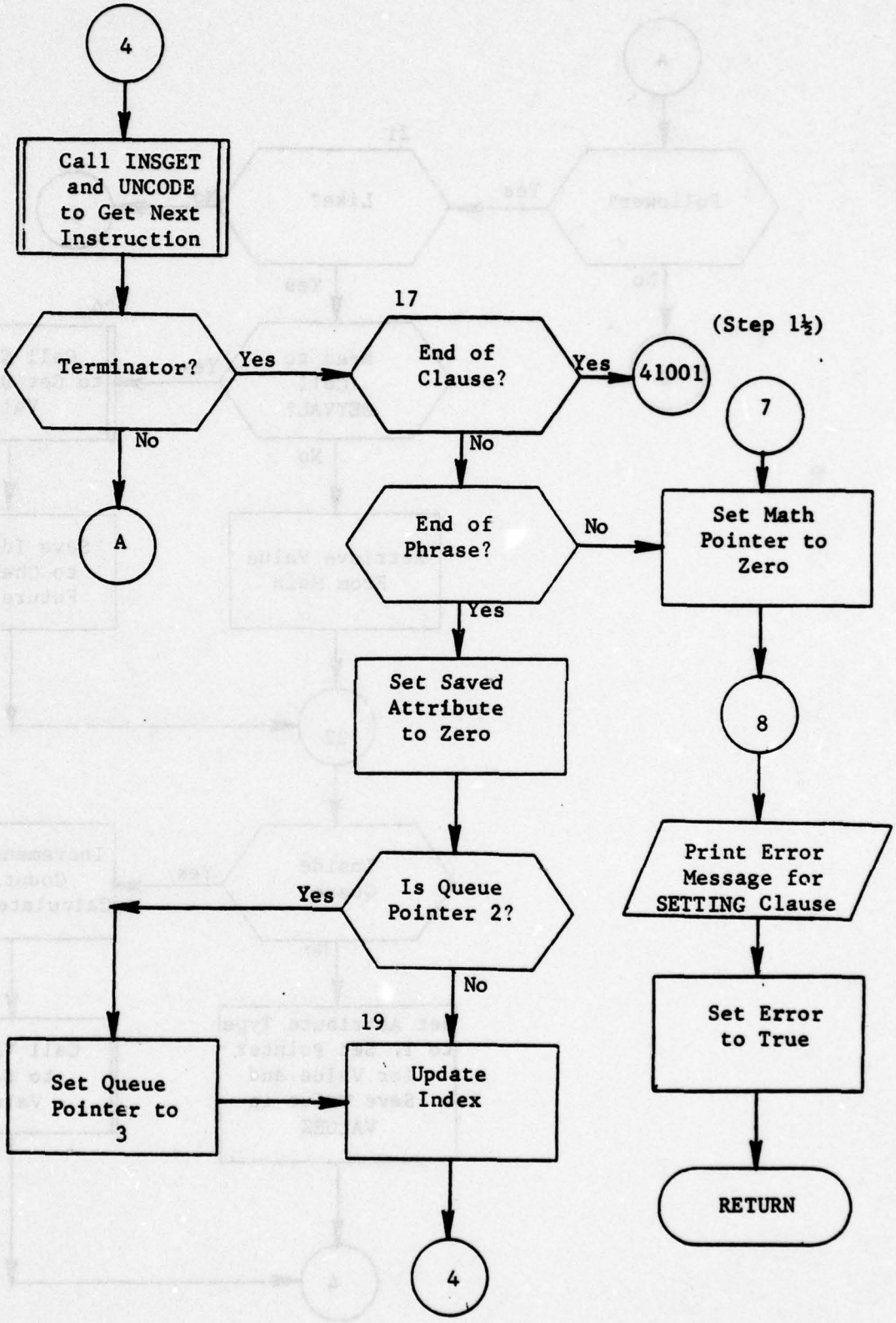


Figure 43. (Part 2 of 10)

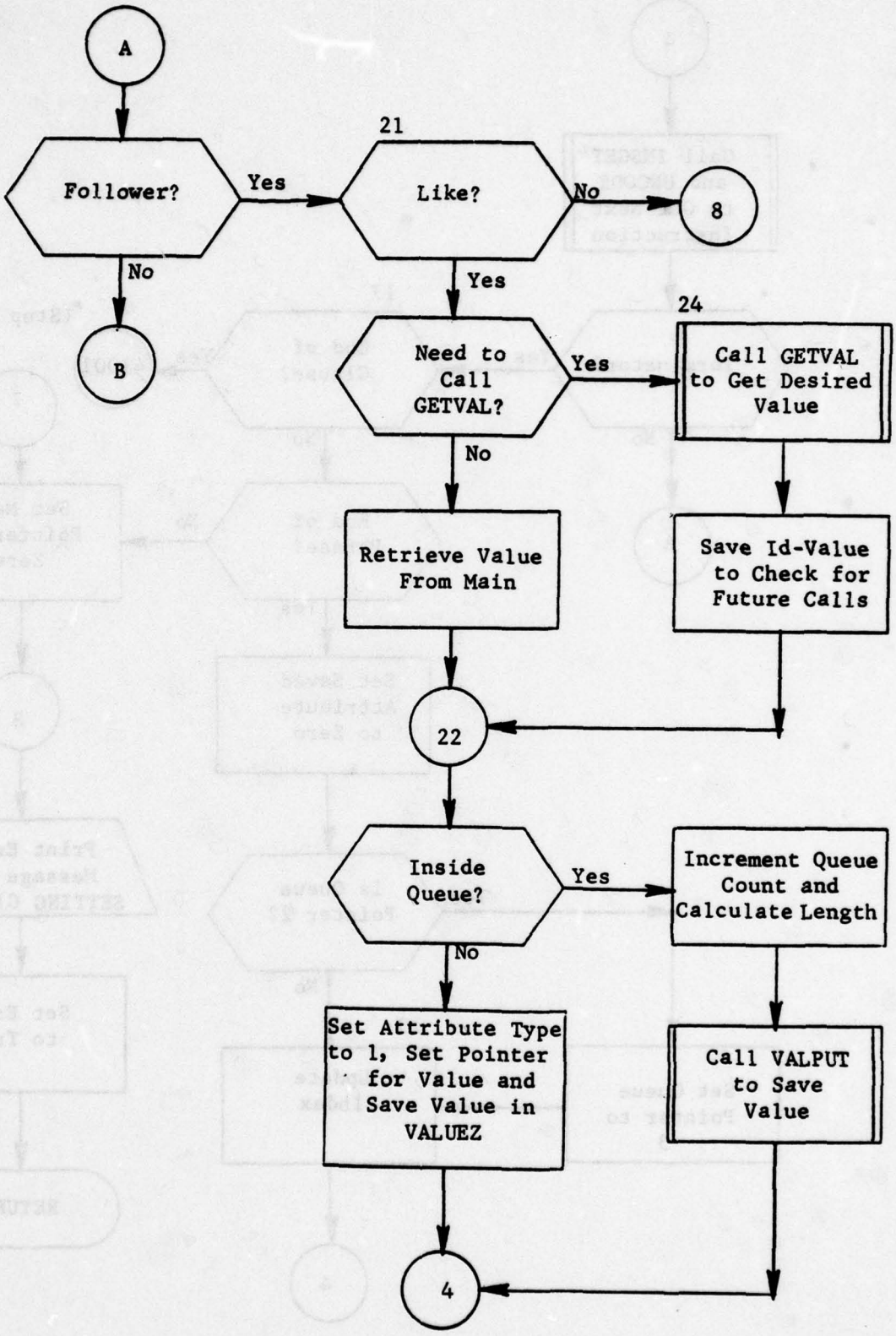


Figure 43. (Part 3 of 10)

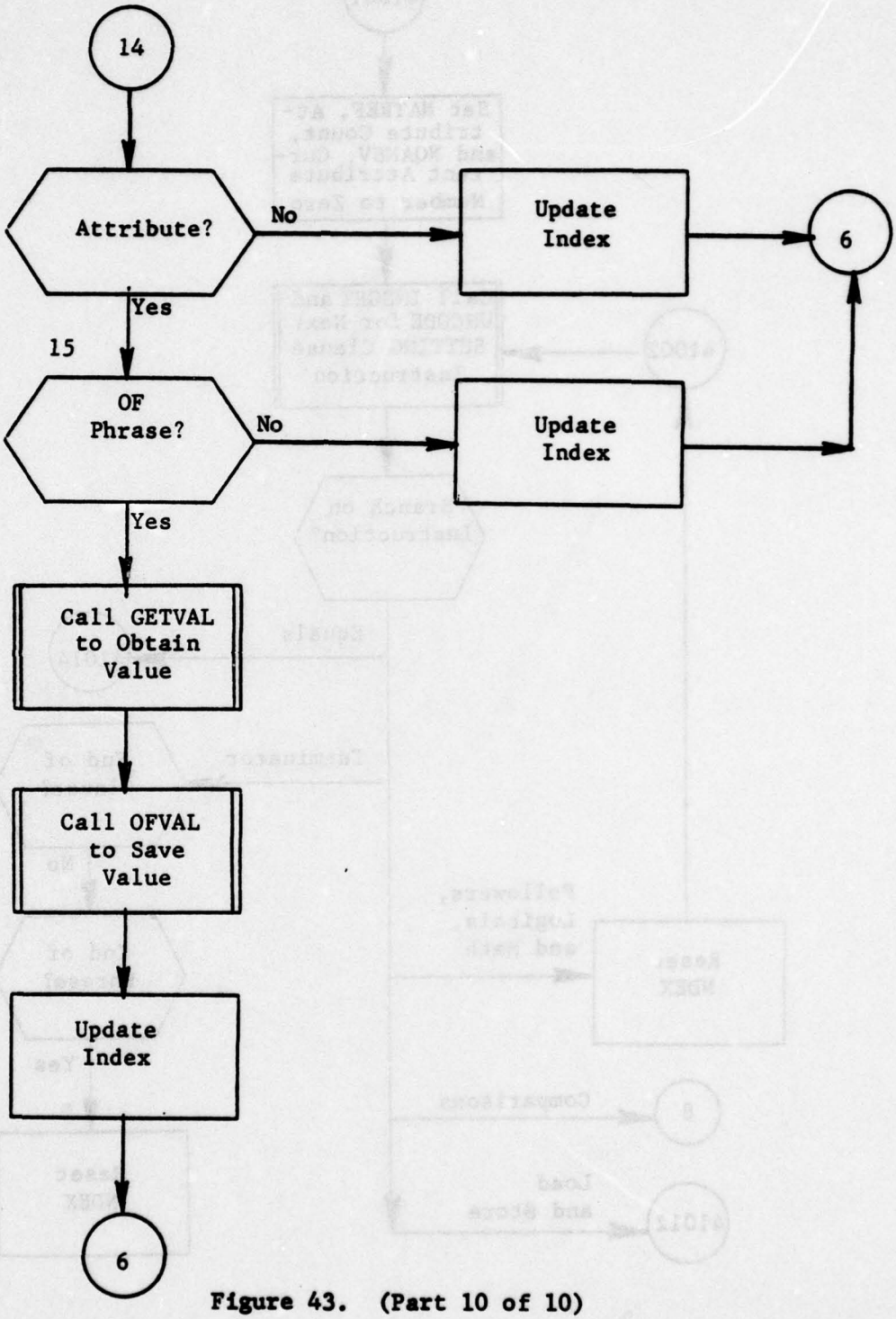


Figure 43. (Part 10 of 10)

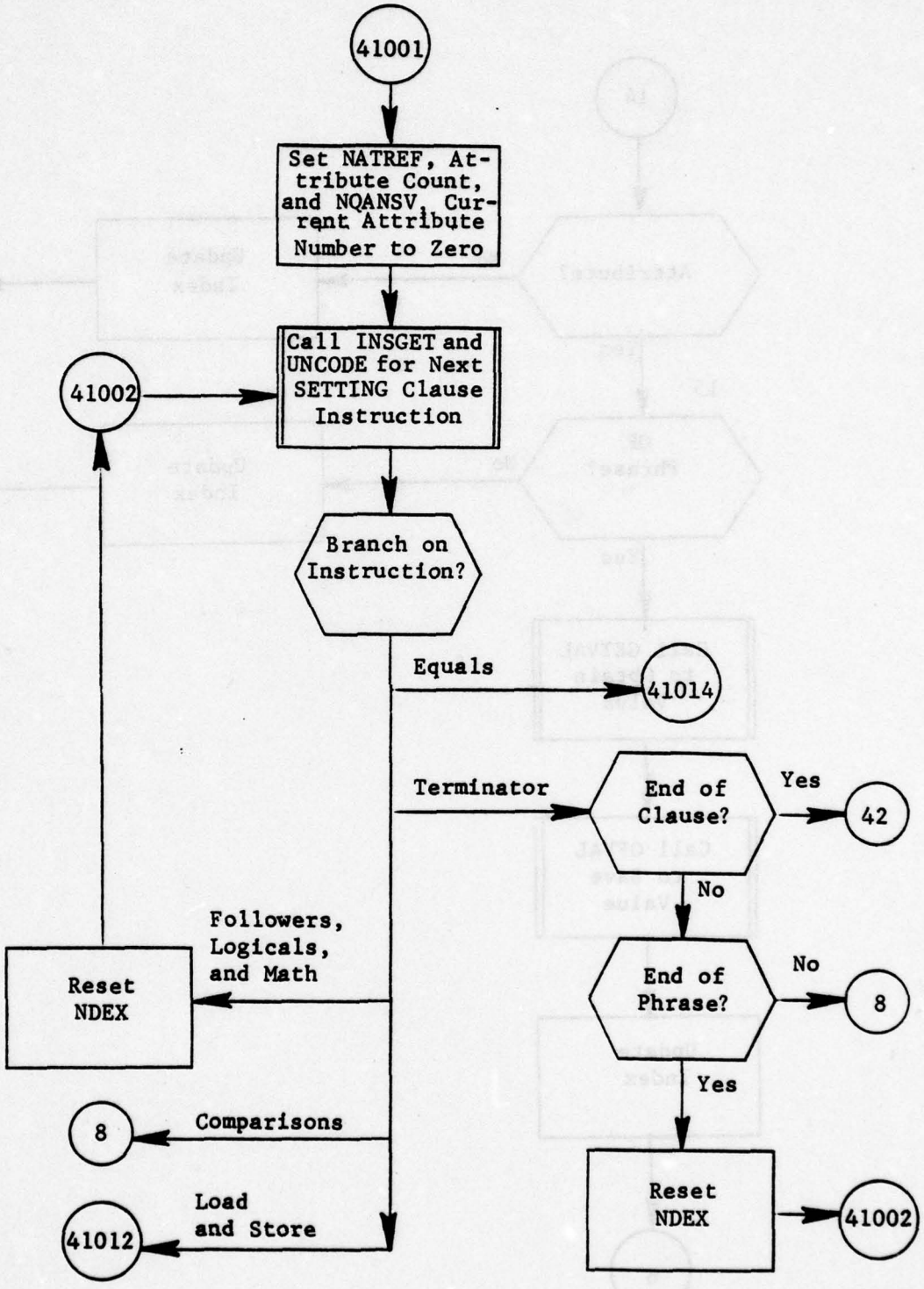


Figure 43.1. Subroutine CHANGE: Step One and a Half (Part 1 of 6)

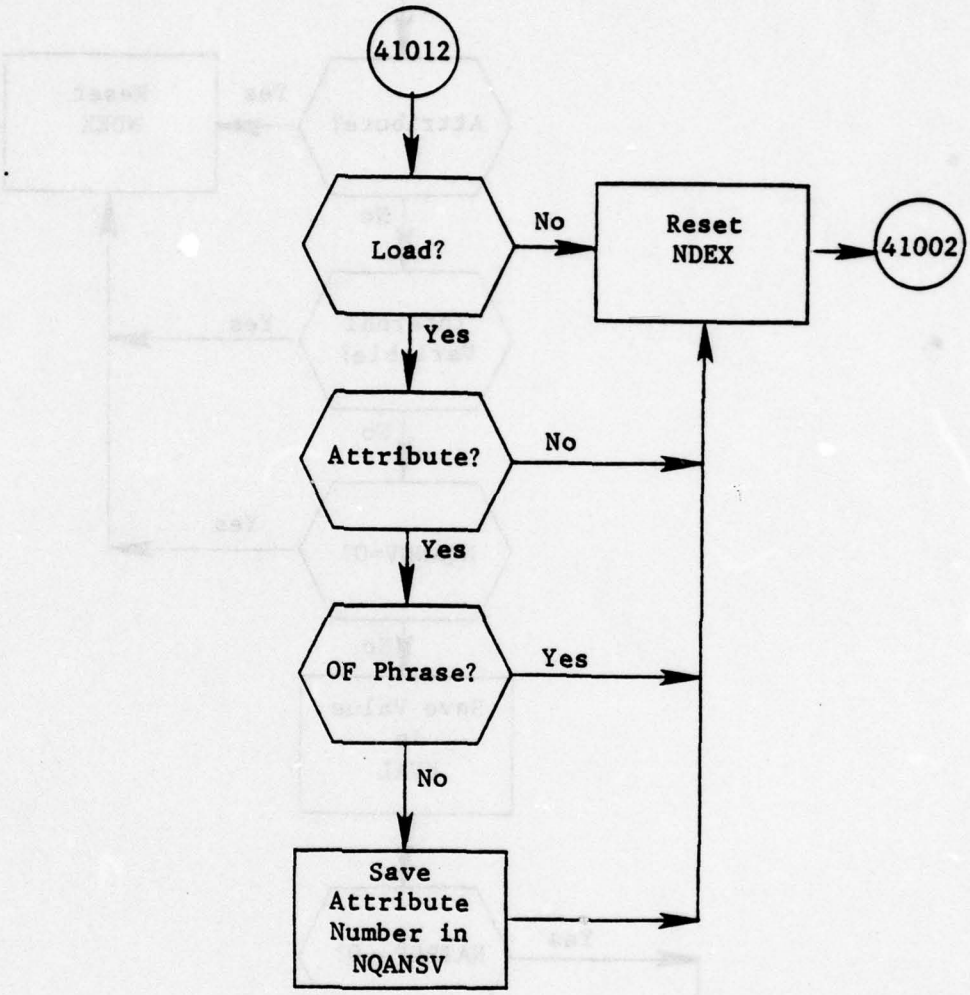


Figure 43.1 (Part 2 of 6)

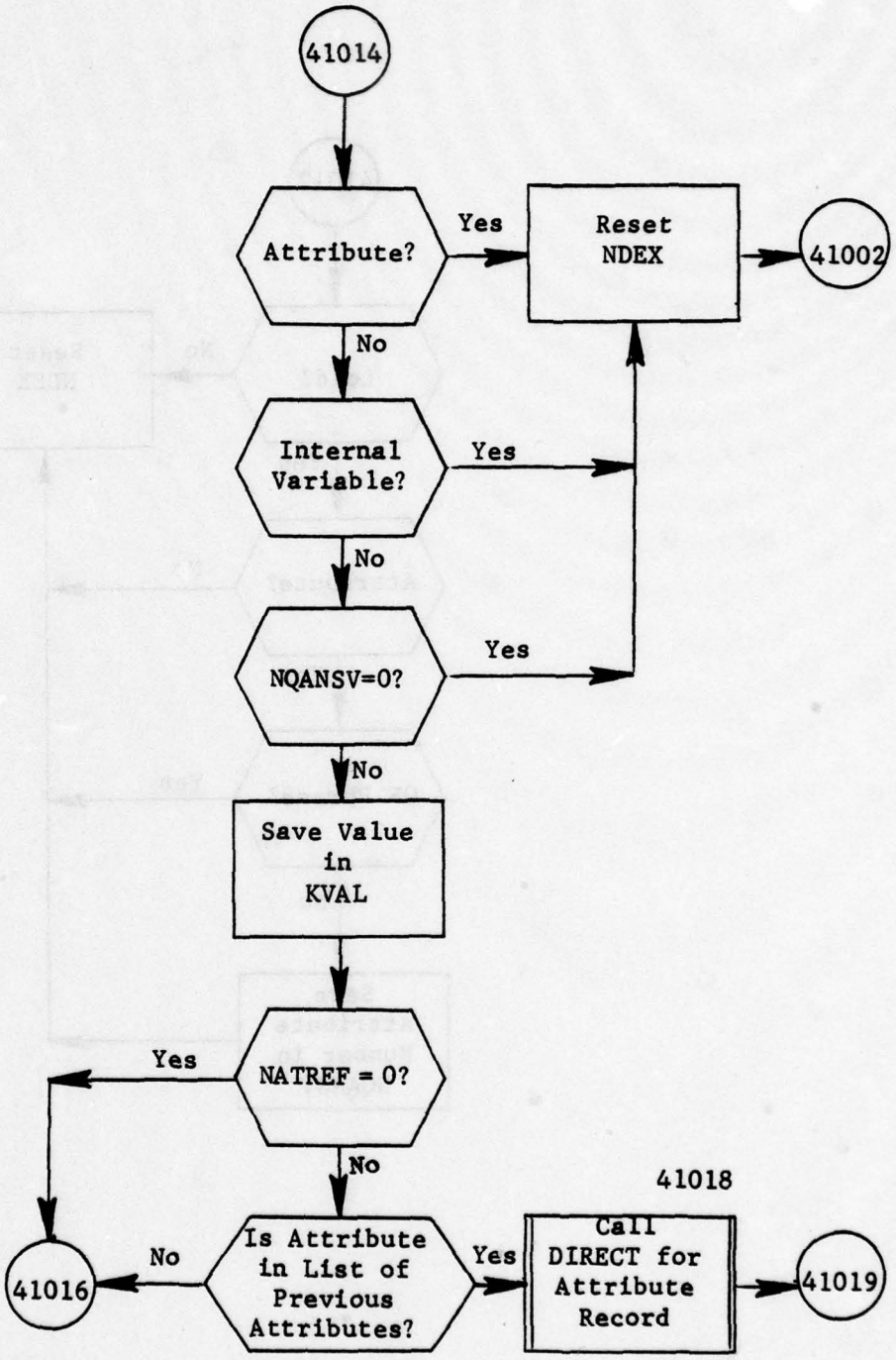


Figure 43.1. (Part 3 of 6)

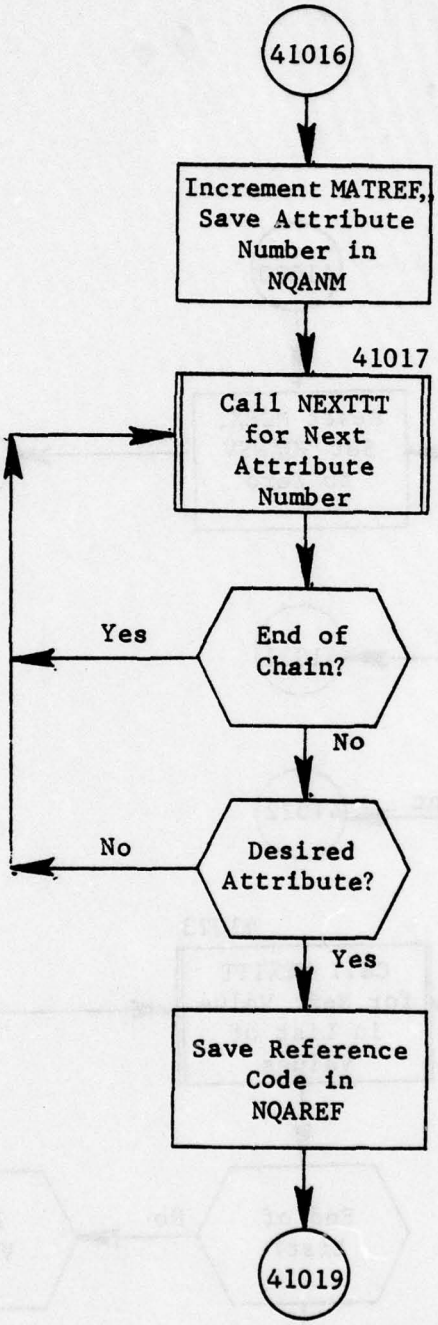


Figure 43.1. (Part 4 of 6)

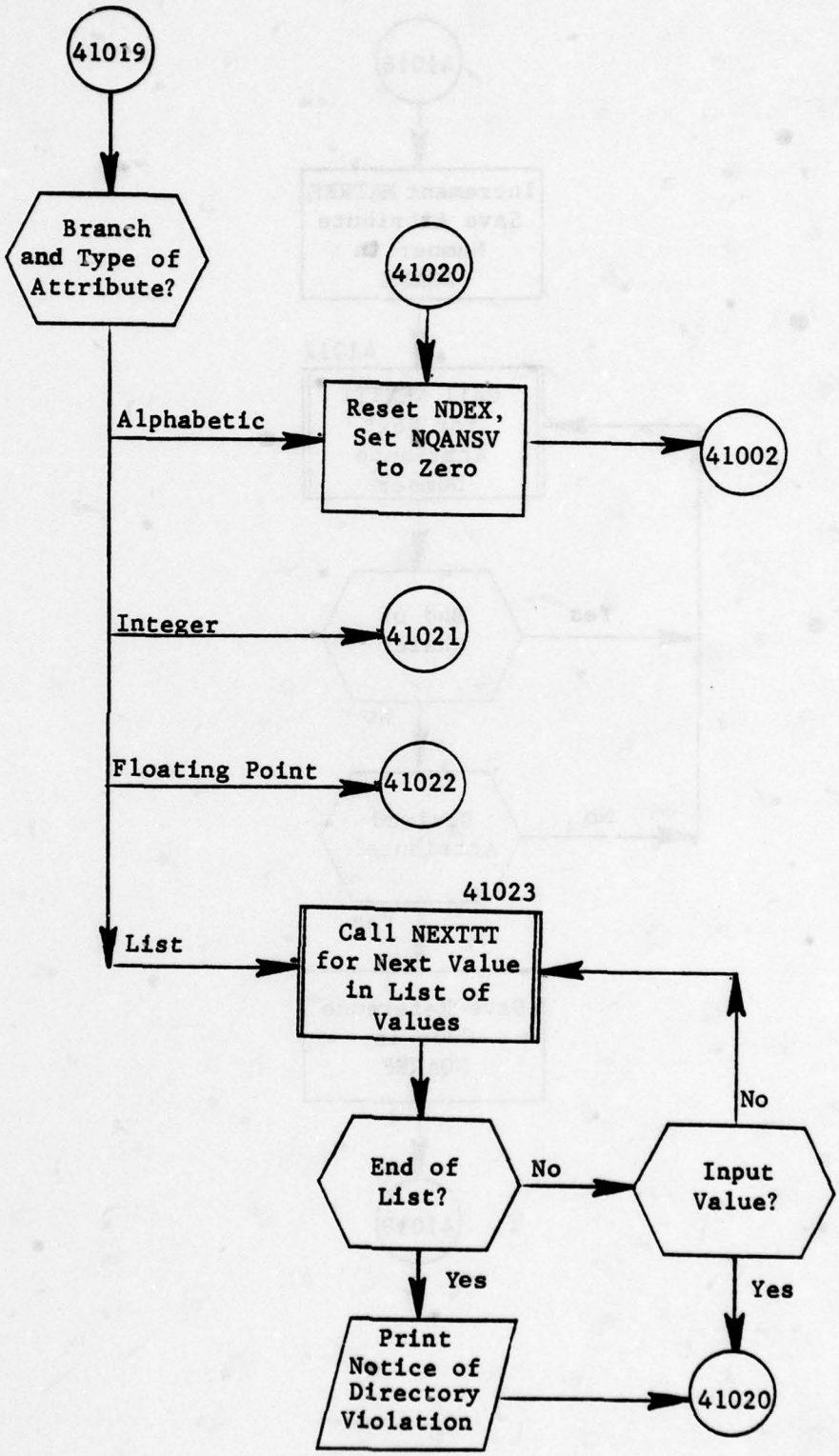


Figure 43.1. (Part 5 of 6)

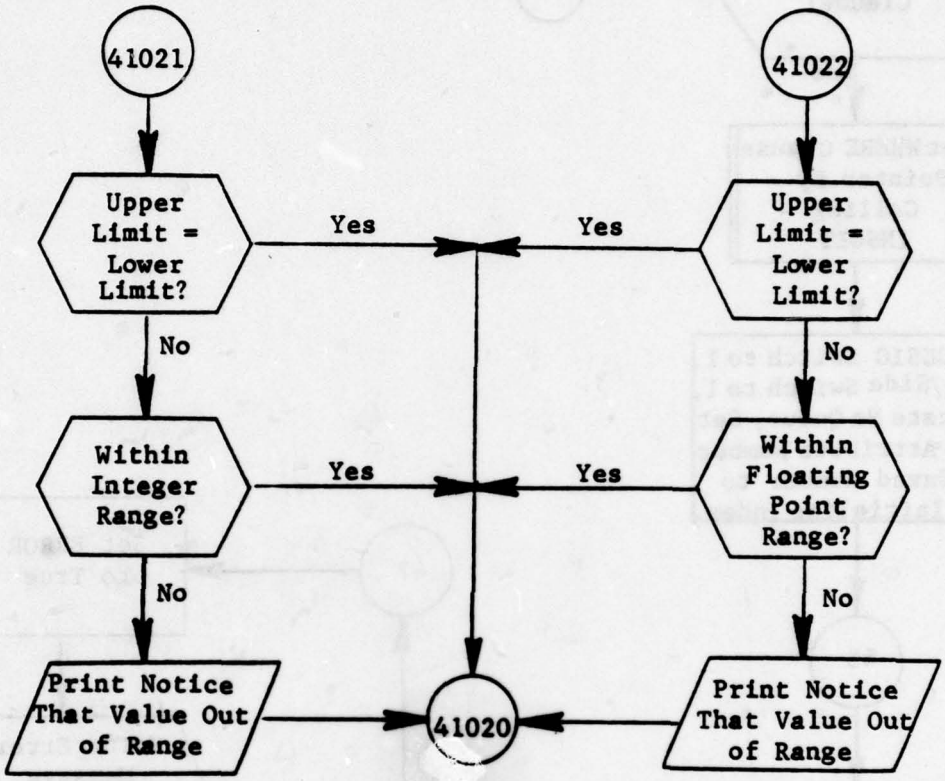


Figure 43.1. (Part 6 of 6)

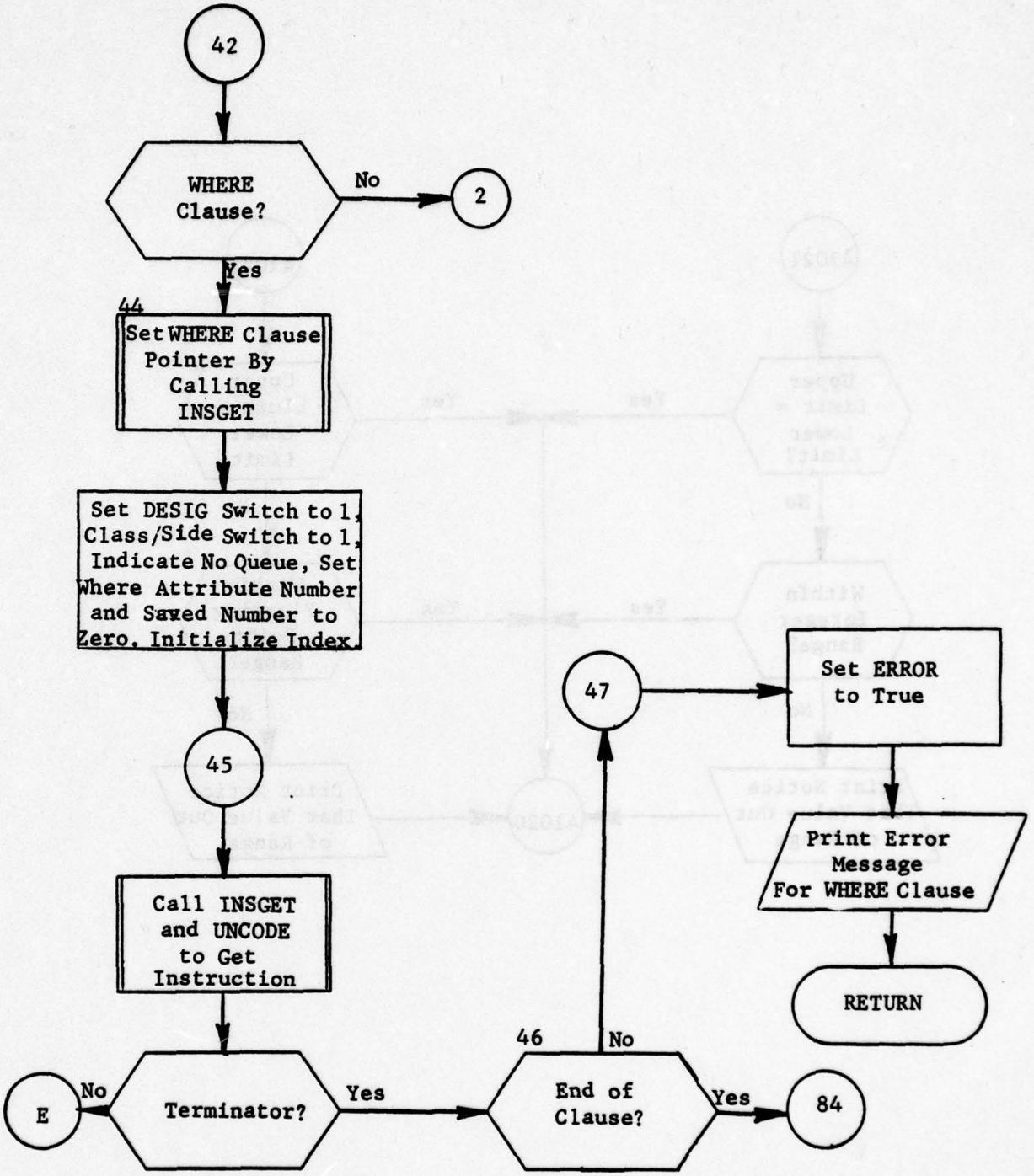


Figure 44. Subroutine CHANGE: Step Two (Part 1 of 9)

Step Three

The two attributes lists (ATNUMB and WHATNB) are now combined. Also it is determined if both clauses contained a queued set of attributes (see figure 45).

Step Four

The list of attributes (ATNUMB) is now used in an attempt to build a list of record types. The ATRIB chain is used to find the attributes in the list and it is determined for each attribute whether it is a single, multiple or control. For a single attribute, the record type it is on is added to the record type list (RTLIST). For multiple attributes (those in the SETTING clause) a list (MLTLST) is kept separately of the record types which contain them. For a control attribute the controlled record is added to the record type list (RTLIST). If the attribute appeared in the SETTING clause, the record type is also added to a special list (CTLLST). A separate list (CRECNM) is also made of the record types whose attributes appear in the SETTING clause as these are the types which will be changed (see figure 46).

Step Five

For each record type in the list (CTLLST) of controlled record types whose attribute was in the SETTING clause, PRIMHD is now called to determine their primary header. These headers are added to the list of record types (RTLIST) (see figure 47).

Step Six

Now the primary header is determined. If a value for CLASS was included, it is used to do this. If no such value was included the record type with the highest number is used. PRIMHD is called to determine the primary header. The primary chains down from the primary header are now checked against the multiple attribute record list. Any found are included in the record type list (RTLIST) (see figure 48).

Step Seven

First LINKUP is called to complete the record type list (RTLIST). The lowest level record in the list is now checked to be sure it is in the list of record types to be changed (CRECNM). SETSCH is now called to build the retrieval scheme. The retrieval scheme is used to determine the retrieval order of the record types which are to be changed and they are placed in the CHORD list in this order (see figure 49).

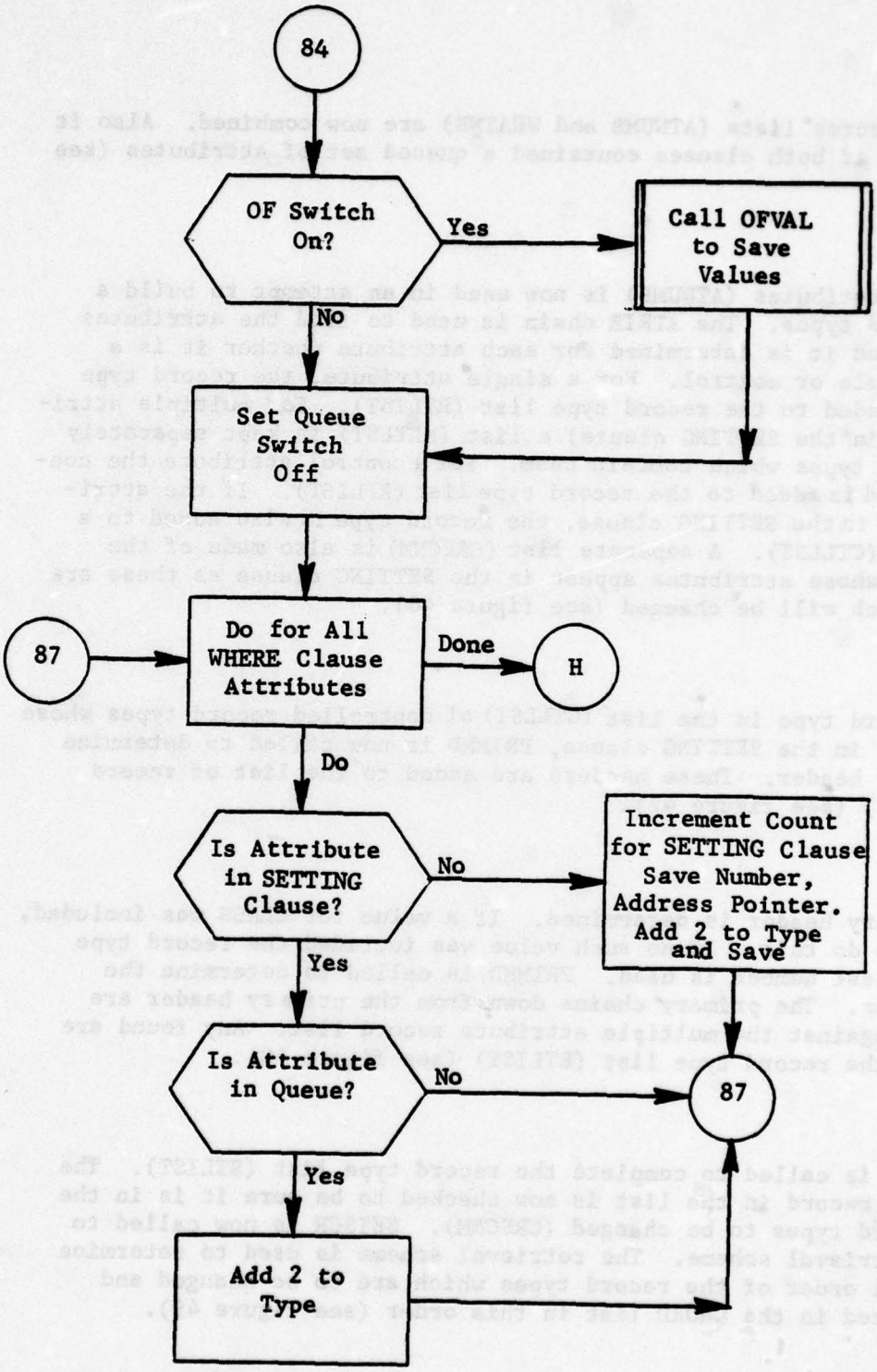


Figure 45. Subroutine CHANGE: Step Three (Part 1 of 2)

4.8.1 Subroutine DESSCH

PURPOSE: To build a DESIG driven retrieval scheme

ENTRY POINTS: DESSCH

FORMAL PARAMETERS: LSTLST: Input list of record type numbers
LSTLN: Number of types in list

COMMON BLOCKS: C10, C20, C30, OOPS, PRINSP, SCHEME, SCRTCH

SUBROUTINES CALLED: NEXTTT

CALLED BY: CHANGE

Method:

This subroutine builds a special retrieval scheme driven by the attribute DESIG. The format of the scheme is a series of word pairs as outlined in table 16. After the invariable first instruction is created, the process begins by searching chains of which the TARGET record is master. For each such search that results in finding a record type in the record type list (LSTLST), a Chain Next instruction is added. The record types identified are placed in an open ended list (MIST). For each record type in MIST, the chains of which it is detail are searched for matches in LSTLST. Each match causes a chain head command to be inserted in the scheme and a new type added to the list. When all types have been processed, the return instruction is added.

Subroutine DESSCH is illustrated in figure 51.

Table 16. DESIG Driver Retrieval Scheme

WORD 1 OF THE INSTRUCTION PAIR	WORD 2 OF THE INSTRUCTION PAIR	DESCRIPTION
1	1	Instructs NXTDES to retrieve the target using the input value for DESIG
2	Chain Name	Instructs NXTDES to call NEXTTT for the chain names
3	Chain Name	Instructs NXTDES to call HEAD for the chain named
4	1	Instructs NXTDES to reset POINT to 1 and return

ACKNOWLEDGMENT

This documentation was prepared under the direction of the Chief for Military Studies and Analysis, CCTC, in response to a requirement of the Studies, Analysis, and Gaming Agency, Organization of the Joint Chiefs of Staff. Technical support was provided by System Sciences, Incorporated under Contract Number DCA100-75-C-0019. Change set two was prepared under Contract Number DCA100-78-C-0035.

Section		Page
	7.3 Output	573
	7.4 Concept of Operation	573
	7.5 Identification of Subroutine Functions	573
	7.6 Common Blocks	573
	7.7 Subroutine ENTMOD	575
8.	EXTERNAL INTERFACE MODULE (EIM)	581
	8.1 Purpose	581
	8.2 Input	581
	8.3 Output	581
	8.4 Concept of Operation	581
	8.5 Identification of Subroutine Functions	581
	8.5.1 Subroutine SIDAC	581
	8.5.2 Subroutine TABBLE	589
	8.5.3 Subroutine BLDOTH	589
	8.5.4 Subroutine PLOTDATA	589
	8.5.5 Subroutine PLOTIT	589
	8.6 Common Blocks	589
	8.7 Subroutine ENTMOD	594
	8.7.1 Subroutine CONVLL	597
	8.8 Subroutine BLDOTH	599
	8.8.1 Subroutine XEDEFN	628
	8.9 Subroutine PLOTDATA	633
	8.9.1 Subroutine PICS	651
	8.9.2 Subroutine PROJCT (Entry PROJCT) (Entry PROJT2)	655
	8.10 Subroutine SIDAC	658
	8.11 Subroutine TABBLE	662
	8.12 Subroutine PLOTIT	676
	8.12.1 Subroutine FNDSRT	676.5
	8.12.2 Subroutine INTRPL	676.8
	8.12.3 Subroutine PLBLOFF	676.10
	8.12.4 Subroutine PLOTINIT	676.14
	8.12.5 Subroutine SUBPLOT	676.19
	8.12.6 Subroutine SUBREAD	676.34
9.	GENERAL UTILITIES	677
	9.1 Purpose	677
	9.2 Subroutine ABORT	683
	9.3 Subroutine ATFNDR	685
	9.4 Function ATN2PI	693
	9.5 Function AZMUTH	695
	9.6 Subroutine CINSGET	697
	9.7 Function DIFFLONG	699
	9.8 Function DISTF	701
	9.9 Subroutine DOTLINE	703

Section Page

9.10 Subroutine FINDCLAS..... 705
9.11 Subroutine FINDSIDE..... 707
9.12 Subroutine FORMAK..... 709
9.13 Function GETCLOCK..... 711
9.14 (Deleted)
9.15 Subroutine GETNXT..... 715
9.16 Subroutine GETSTR..... 719
9.17 Subroutine GETTAR..... 726
(Entry GETTAR)
(Entries FNDTAR and GETDES)
9.18 Function GLOG..... 737
9.19 Subroutine HOUSKEEP..... 739
9.20 Function IGET..... 741
9.21 Function IGETHOB..... 743
9.22 Subroutine INTERP..... 745
9.23 Subroutine INTRPGC..... 748
9.24 Subroutine INTPIECE..... 753
9.25 Subroutine IORFL..... 755
9.26 Subroutine IPUT..... 757
9.27 Subroutine ISOFF..... 759
9.28 Function ITLE..... 761
9.29 (Deleted)
9.30 Function KEYMAKE..... 765
9.31 (Deleted)
9.32 Subroutine LINKUP..... 769
9.33 Subroutine LREORDER..... 779
9.33.1 Subroutine LUNCH..... 780.1
9.34 Subroutine MAPEDGE..... 781
9.35 Subroutine MISDATA 783
9.36 Subroutine OFVAL..... 786
9.37 Subroutine ORDER..... 790
9.38 (Deleted)
9.39 Subroutine PIECEIT..... 794
9.40 Subroutine PIECENUM..... 799
9.41 (Deleted)
9.42 Subroutine PRIMHD..... 803
9.43 Subroutine PSREC..... 805
(Entry PSREC)
(Entry PSPUT)
(Entry PSRWD)
(Entries XSORT and PSNXT)
9.44 Function RANGER 811
9.45 Subroutine REORDER..... 814
9.46 Subroutine SETORD..... 816
(Entry SETORD)
(Entry RESORD)
9.47 Subroutine SETSCH..... 820
9.48 Function SLOG..... 830
9.49 Subroutine SORTIT..... 832
9.50 Function SSKPC..... 836

Section		Page
	9.51 (Deleted)	840
	9.52 Subroutine SVTP (Entry FILLOD) (Entry FILSAV)	842
	9.53 (Deleted)	848
	9.54 (Deleted)	851
	9.55 Subroutine TIMEME	853
	9.56 Function TOFM	856
	9.57 Subroutine UNCODE	858
	9.58 Subroutine VALFND	860
	9.59 Function VALTAR	870
	9.60 (Deleted)	872
	9.61 Subroutine XMATH	874
	9.62 Subroutine XWHERE	878
	9.63 Function ZTAN	888
APPENDIXES		
A.	COP External Common Blocks	891
B.	Executable Job Control Language (JCL) QUICK System	895
C.	PERFORM Program	909
DISTRIBUTION		923
DD Form 1473		925

Figure		Page
125	Subroutine XEDEFN	629
126	Subroutine PLOTDATA	634
127	Subroutine PICS	652
128	Subroutine PROJCT	656
129	Subroutine SIDAC	659
130	Subroutine TABBLE	663
130.1	Subroutine PLOTIT	676.1
130.2	Subroutine FNDSRT	676.6
130.3	Subroutine INTRPL	676.9
130.4	Subroutine PLBLOFF	676.11
130.5	Subroutine PLOTINIT	676.15
130.6	Subroutine SUBPLOT	676.21
130.7	Subroutine SUBREAD	676.35
131	Subroutine ABORT	684
132	Subroutine ATFNDR	687
133	Function ATN2PI	694
134	Function AZMUTH	696
135	Subroutine CINSGET	698
136	Function DIFFLONG	700
137	Function DISTF	702
138	Subroutine DOTLINE	704
139	Subroutine FINDCLAS	706
140	Subroutine FINDSIDE	708
141	Subroutine FORMAK	710
142	Function GETCLOCK	712
143	(Deleted)	714
144	Subroutine GETNXT	717
145	Subroutine GETSTR	720
146	Subroutine GETTAR	728
147	Function GLOG	738
148	Subroutine HOUSKEEP	740
149	Function IGET	742
150	Function IGETHOB	744
151	Subroutine INTERP	747
152	Coordinate System for INTRPGC	749
153	Subroutine INTRPGC	752
154	Subroutine INTPIECE	754
155	Subroutine IORFL	756
156	Subroutine IPUT	758
157	Subroutine ISOFF	760
158	Function ITLE	762
159	(Deleted)	764
160	Function KEYMAKE	766
161	(Deleted)	768
162	Subroutine LINKUP	770
163	Subroutine LREORDER	780
163.1	Subroutine LUNCH	780.2
164	Subroutine MAPEDGE	782
165	Subroutine MISDATA	784
166	Subroutine OFVAL	787

Figure		Page
167	Subroutine ORDER	791
168	(Deleted)	793
169	Subroutine PIECEIT	796
170	Subroutine PIECENUM	800
171	(Deleted)	802
172	Subroutine PRIMHD	804
173	Subroutine PSREC	807
174	Function RANGER	812
175	Subroutine REORDER	814
176	Subroutine SETORD	818
177	Subroutine SETSCH	822
178	Function SLOG	831
179	Subroutine SORTIT	833
180	Function SSKPC	838
181	(Deleted)	841
182	Subroutine SVTP	843
183	(Deleted)	849
184	(Deleted)	852
185	Subroutine TIMEME	855
186	Function TOFM	857.1
186.1	(Deleted)	857.5
187	Subroutine UNCODE	859
188	Subroutine VALFND	862
189	Function VALTAR	871
190	(Deleted)	873
191	Subroutine XMATH	875
192	Subroutine XWHERE	879
193	Function ZTAN	889
194	H* Creation From Source	896
195	H* Creation From Object	902
196	Utility Library Creation	906
197	Program PERFORM	912
198	Subroutine READIN	921
199	Subroutine IDENT	921.2

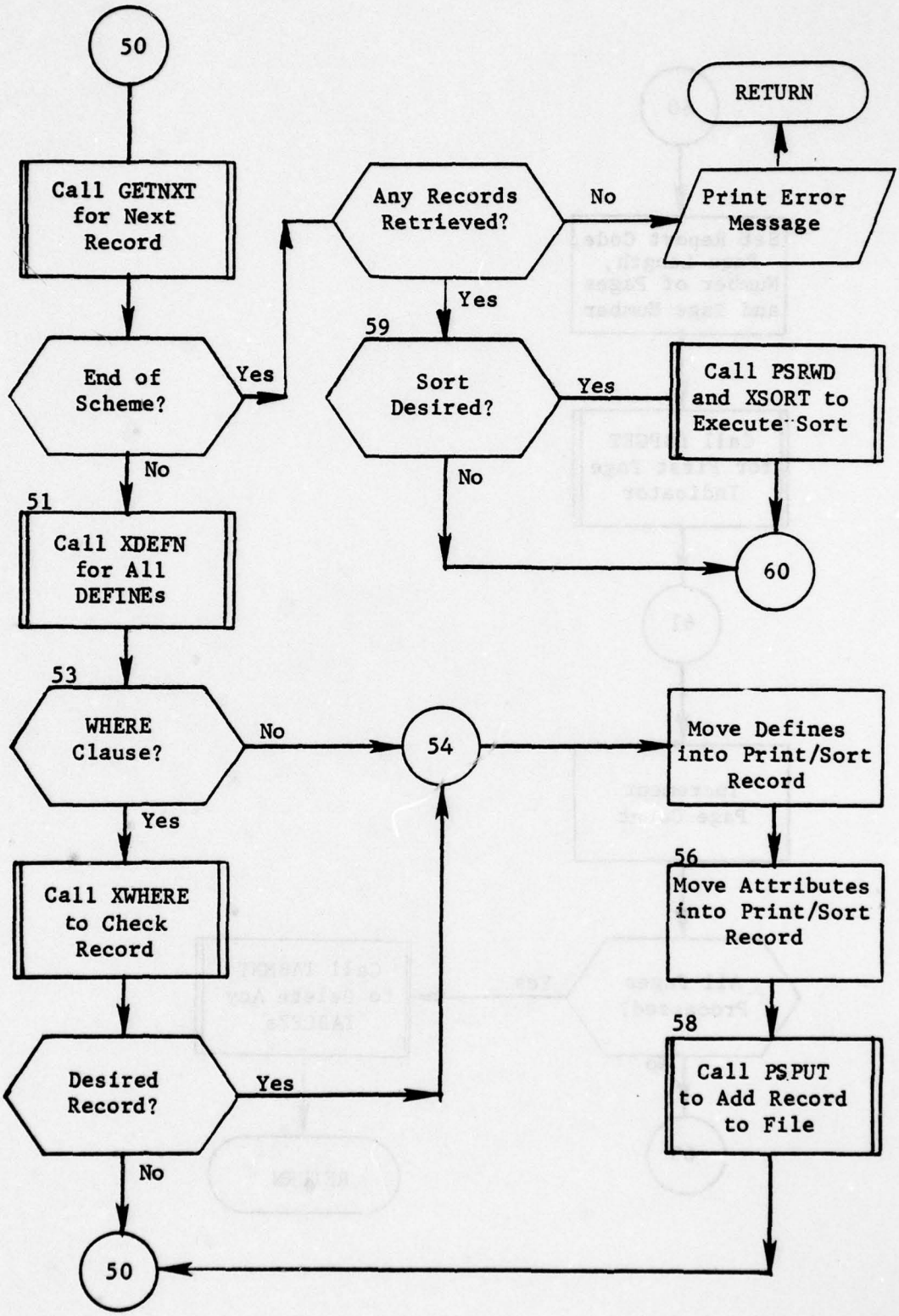


Figure 112. (Part 2 of 2)

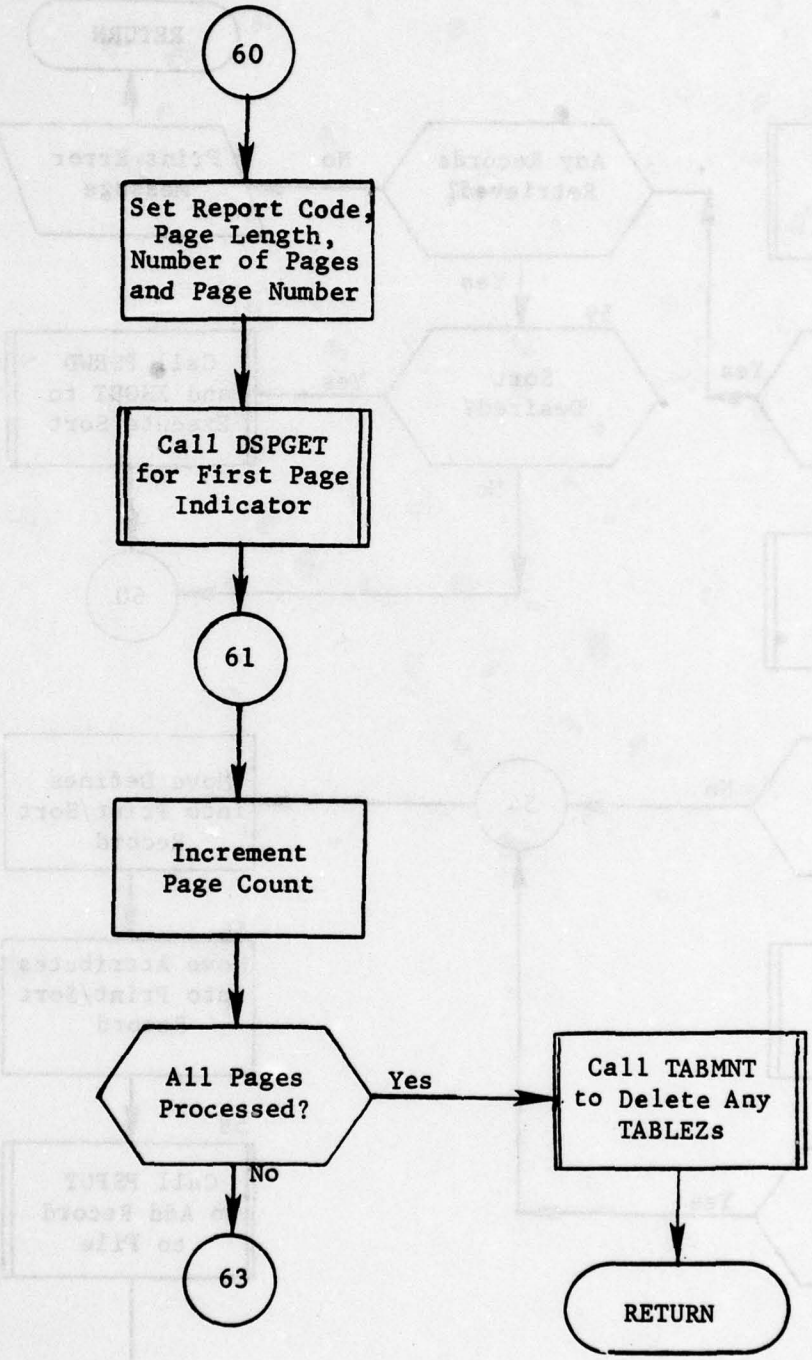


Figure 113. Subroutine PRINCE: Step Five (Part 1 of 8)

SECTION 8. EXTERNAL INTERFACE MODULE (EIM)

8.1 Purpose

The purpose of the EIM is to create output tapes/files which are designed to be input to external processors.

8.2 Input

The output files to be built each have the precondition that all data necessary for the file be present in the data base.

8.3 Output

The output of EIM depends upon the verb and the FILE clause. BUILD FILE TABLE produces a single tape with six subsections. The format of this tape appears in table 21. BUILD FILE SIDAC produces two tapes, one containing BLUE targets, one containing RED targets. The format of the two tapes appears in table 22. BUILD FILE OTHER produces a single tape or file whose format and contents are specified by the user.

The PLOTIT and PLOTDATA verbs produce two tapes. One is a tape suitable for the CALCOMP plotter. The other contains information concerning all points which were not added to the plot tape owing to their being out of range of the plot size.

8.4 Concept of Operation

The EIM first determines which verb caused the call. If the verb was PLOTIT or PLOTDATA, the appropriate routine is executed. If the verb was BUILD, the FILE clause is found along with the special word within the clause. If the special word is SIDAC or TABLE the appropriate subroutines are called to produce the named files. If the special word is OTHER, the BLDOTH subroutine is called to produce the named files. If the special word is OTHER, the BLDOTH subroutine is called to produce the input defined file.

8.5 Identification of Subroutine Functions

8.5.1 Subroutine SIDAC. This subroutine produces the data base assessment tapes (DBASSESS). A preset retrieval scheme for all BLUE targets is set up and executed. Each target retrieved is written out in a modified JAD format. When all BLUE targets have been output, the scheme is modified for RED targets and the RED targets are written onto a separate tape.

Table 21. BUILD FILE TABLE Output File Formats
(Part 1 of 5)

TARGET LIST

Column	Meaning
1-8	'F1TARGET'
9	Side: 1 for Blue; 2 for Red
10-14	Line Count, numeric
15	Blank
16-20	DESIG, alphabetic.
21-24	Blank
25-31	Latitude (LAT), degrees, minutes, seconds
32-39	Longitude (LONG), degrees, minutes, seconds
40-45	NAME, alphabetic
46-49	World Area Code (WACNO), alphabetic
50-55	Bomber Encyclopedia Number (BENO), alphabetic
56-60	Category (CATCODE), numeric
61-62	Country Location (CNTRYL), alphabetic
63-68	Major Complex Number (MAJOR), numeric
69-71	SIOP Table Number, numeric
72-76	Index Number (INDEXNO), numeric
77	Blank
78-80	Complex Number (ICOMPL), numeric
81-90	Blank

Table 21. (Part 2 of 5)

VEHICLE CHARACTERISTICS LIST

Column	Meaning
1-7	'F1VEHIC'
8	Blank
9	Side: 1 for Blue; 2 for Red
10-14	Line count, numeric
15	'1'
16-18	Blank
19-20	SAGA Plane type code
21-55	Blank
56-59	CEP in tems of feet
60-69	Blank
70-75	TYPE, alphabetic
76-90	Blank

WEAPON CHARACTERISTICS LIST

Column	Meaning
1-8	'F1WEAPON'
9	Side: 1 for Blue; 2 for Red
10-14	Line count
15-17	Blank
18-19	Warhead type (Line Count * 10 plus 1 for ASM 0 for all others)
20	0 = Bomb, 1 = ASM, 2 = DECOY
21-37	Blank
38-43	Warhead yield in kilotons
44-46	Fission to Fusion percentage (FFRAC*100)
47-90	Blank

Table 21. (Part 3 of 5)

MISSILE BASE LIST

Column	Meaning
1-8	'F1MIBASE'
9	Side: 1 for Blue; 2 for Red
10-14	Line Count
15	Blank
16-20	Line Count
21.	Blank
22-28	Latitude (LAT) degrees, minutes, seconds
29-36	Longitude (LONG) degrees, minutes, seconds
37	Blank
38-41	Vulnerability Number (VULN1) alphabetic
42-43	Type Count
44-45	Blank
46-47	'1/'
48-49	Number per site (NMPSIT), numeric
50	Blank
51	H if VN greater than or Equal to 20 S otherwise
52	Blank
53	1 if column 51 is H or if 51 is S and NOALER Equal NMPSIT Otherwise = 2
54-59	Blank
60-65	NAME, alphabetic
66-69	Blank
70-71	Country Location (CNTRYL), alphabetic
72-74	Blank
75-80	TYPE, alphabetic
81-84	Blank
85-90	Bomber Encyclopedia Number (BENO), alphabetic

Table 21. (Part 4 of 5)

BOMBER BASE LIST

Column	Meaning
1-6	'F1BASE'
7-8	Blank
9	Side: 1 for Blue; 2 for Red
10	Blank
11-14	Line Count
15	Blank
16-20	Index Number (INDEXNO), numeric
21	Blank
22-28	Latitude (LAT), degrees, minutes, seconds
29-36	Longitude (LONG), degrees, minutes, seconds
37	Blank
38	1 for SLBM, 2 for LRA, 3 for TAC, 7 for all others (from FUNCTI)
39	Blank
40	'X'
41-43	Blank
44	'X' for tanker blank for all others
45-59	Blank
60-65	NAME, alphabetic
66-69	Blank
70-71	Country Location (CNTRYL), alphabetic
72-84	Blank
85-90	Bomber Encyclopedia Number (BENO), alphabetic

Table 21. (Part 5 of 5)

OFFENSIVE RECOVERY BASE LIST

Column	Meaning
1-7	'F1RECBS'
8	Blank
9	Side: 1 for Blue; 2 for Red
10-14	Line Count
15	Blank
16-20	DESIG, alphabetic
21-23	Blank
24-30	Latitude (LAT), degrees, minutes, seconds
31-38	Longitude (LONG), degrees, minutes, seconds
39	Blank
40-45	NAME, alphabetic
46-49	World Area Code (WACNO), alphabetic
50-55	Bomber Encyclopedia Number (BENO), alphabetic
56-60	Category Code (CATCODE), numeric
61-62	Country Location (CNTRYL), alphabetic
63-68	Major Complex Number (MAJOR), numeric
69-70	'00'
71-75	Index Number (NDEXNO), numeric
76	Blank
77-80	Capacity (CAPACITY), numeric
81-90	Blank

Table 22. BUILD FILE SIDAC Output File Format
(Part 1 of 2)

Column	Meaning
1-5	Category code, (CATCODE) numeric
6-9	World Area Code (WACNO) alphabetic
10-15	Bomber Encyclopedia Number (BENO) alphabetic
16-20	Blank
21-26	Name (NAME) alphabetic
27-58	Blank
59-64	Major Complex Number (MAJOR) numeric
65-88	Blank
89-94	Minor Compound Number (MINOR) numeric
95-118	Blank
119-125	Latitude (LAT) degrees, minutes, seconds
126-133	Longitude (LONG) degrees, minutes, seconds
134-137	Blank
138-139	Country Location (CNTRYL) alphabetic
140-147	Blank
148-149	Country Owner (CNTRYO) alphabetic
150-155	Blank
156-159	Severe vulnerability (VULN1) VNTK
160-163	Moderate vulnerability (VULN2) VNTK
164-167	"03PO"
168-190	Blank
191-198	Capacity (POP*10). This quality is zero for all non-U/I targets
199-205	Blank
206-208	Radius (RADIUS*10) numeric - tenth of nautical miles. This quantity is zero for all non-U/I targets
209-283	Blank
284-286	SIOP Table Number, numeric
287-288	Blank

Table 22. (Part 2 of 2)

Column	Meaning
289-293	DESIG, alphabetic
294	Blank
295-300	TYPE, alphabetic
301-303	Blank
304-305	ICLASS, numeric
306	1 for Blue targets; 2 for Red targets
307-318	Blank
319	Region (IREG), numeric
320	SAGA region. This quantity is IREG +1 unless country location is US or AK in which case it is IREG
321-335	Blank
336	Record Mark

8.5.2 Subroutine TABBLE. This subroutine produces the table file. The following tables are produced:

- o Target list
- o Vehicle characteristics list
- o Weapon characteristics list
- o Missile base list
- o Bomber base list
- o Offensive recovery base list

For each list a preset retrieval scheme is executed and the records retrieved are written onto the output tape.

8.5.3 Subroutine BLDOTH. This subroutine builds an output file according to user specified formats. This option is very similar in concepts to the REPORT module. First, all input clauses are scanned for attributes and a retrieval scheme is built. Next, all DEFINEs are placed in proper execution order and DEFINE variable execution tables are built. Next, the WHERE clause is scanned for DEFINEs and altered to handle them properly. The sort scheme is now created. Next, the retrieval scheme is executed and a file built of the resultant records and this file is sorted. Finally, the FORMAT clause is executed to produce the desired file (see figure 116).

8.5.4 Subroutine PLOTDATA. This subroutine builds an output plot tape. Four types of geographic data may be plotted:

- o Penetration corridors
- o Depenetration corridors
- o Refuel points
- o Recovery bases

For each desired set of data a preset retrieval scheme is used to retrieve the appropriate data. For each data record retrieved, the coordinates (LAT, LONG) are converted and/or scaled to the desired map characteristics and are processed for the plotter.

8.5.5 Subroutine PLOTIT. This subroutine builds an output plot tape. The plots produced are of bomber or tanker sorties. User selected sorties are retrieved and their events are plotted in sequence with special symbols used to distinguish events (see CSM UM 9-77, Volume I).

8.6 Common Blocks

The common blocks internal to EIM are listed in table 23.

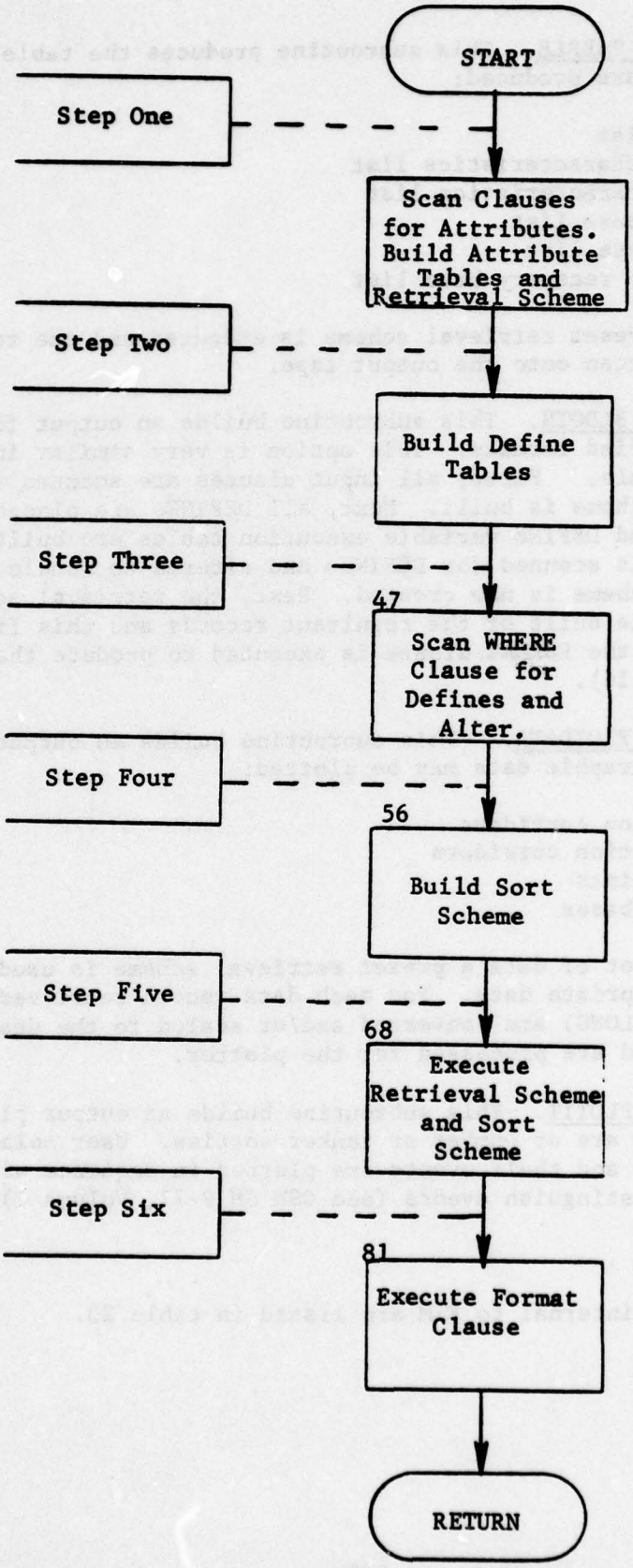


Figure 116. Subroutine BLDOTH: Macro Flow

8.9 Subroutine PLOTDATA*

PURPOSE: To plot geographic data

ENTRY POINTS: PLOTDATA

FORMAL PARAMETERS: None

COMMON BLOCKS: C15, C20, C30, OOPS, PLTPRO, PLTSPE, PRINSP, SCHEME, TAPES, ZEES

SUBROUTINES CALLED: GETNXT, HOUSKE, HOUSK2, INSGET, NEWPEN, NEXTTT, PIECEI, PIECEN, PIECE1, PIECE3, PIECE4, PICS, PROJCT, PROJT2, UNCODE

CALLED BY: ENTMOD (EIM)

Method:

First the SETTING clause is read for changes to the normal settings of the plot selection factors (SCALE, LAT, LONG, etc.). Next the plot tape is initialized through calls to PIECEIT and HOUSKEEP. The general method from then on is for each of the types of plots desired: penetration corridors, depenetration corridors, refuel points and recovery bases. A preset retrieval scheme is read from subroutine data (arrays: SCHA, SCHB, SCHC and SCHD) into common block SCHEME. The value for SIDE is inserted and GETNXT is used to retrieve each desired item for the plot. The coordinates are then adjusted by PICS or PROJT2 and PIECEIT or PIECENUM called to perform the plot.

Subroutine PLOTDATA is illustrated in figure 126.

*Main routine of overlay PLOTIT

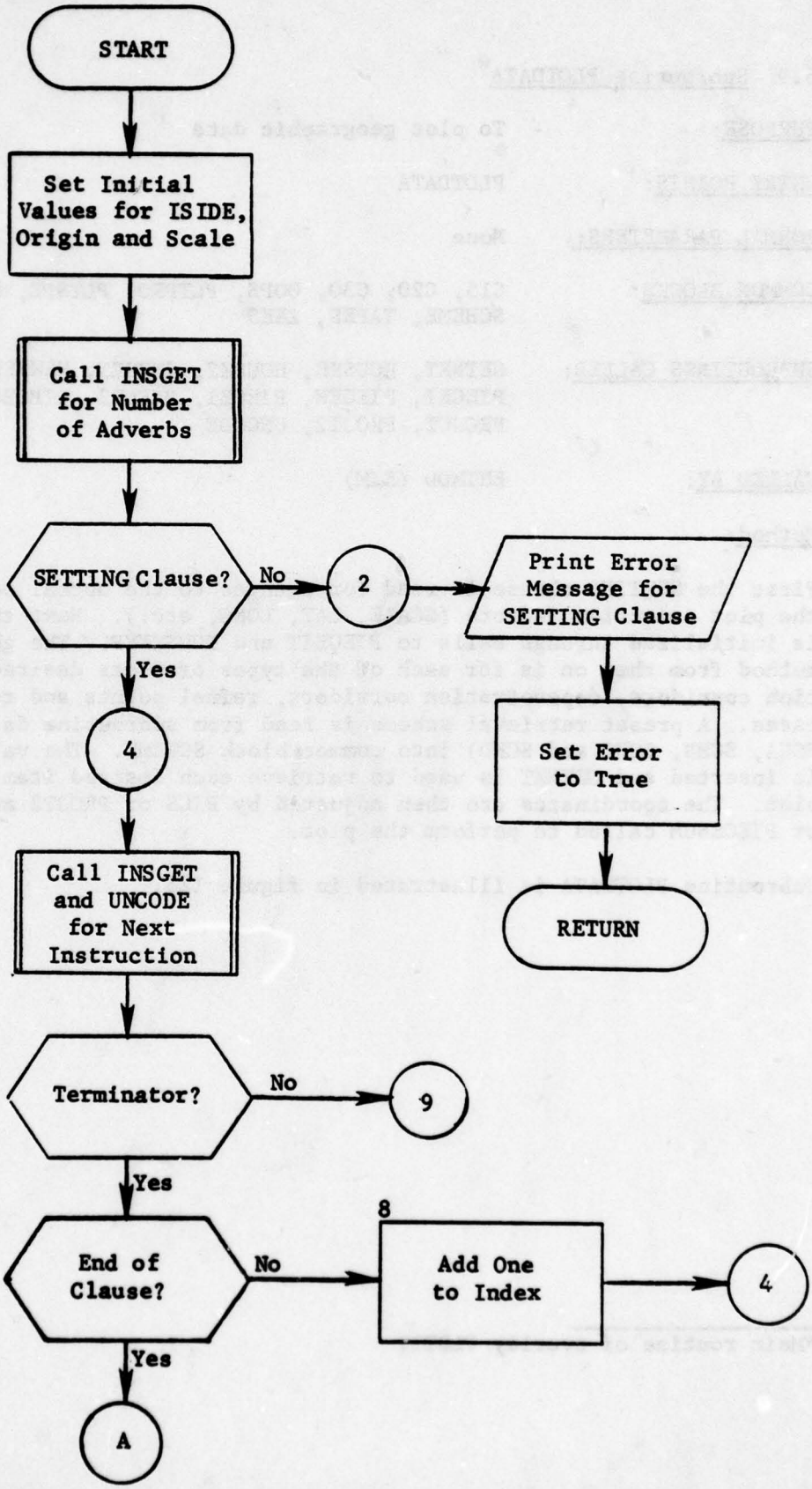


Figure 126. Subroutine PLOTDATA (Part 1 of 17)

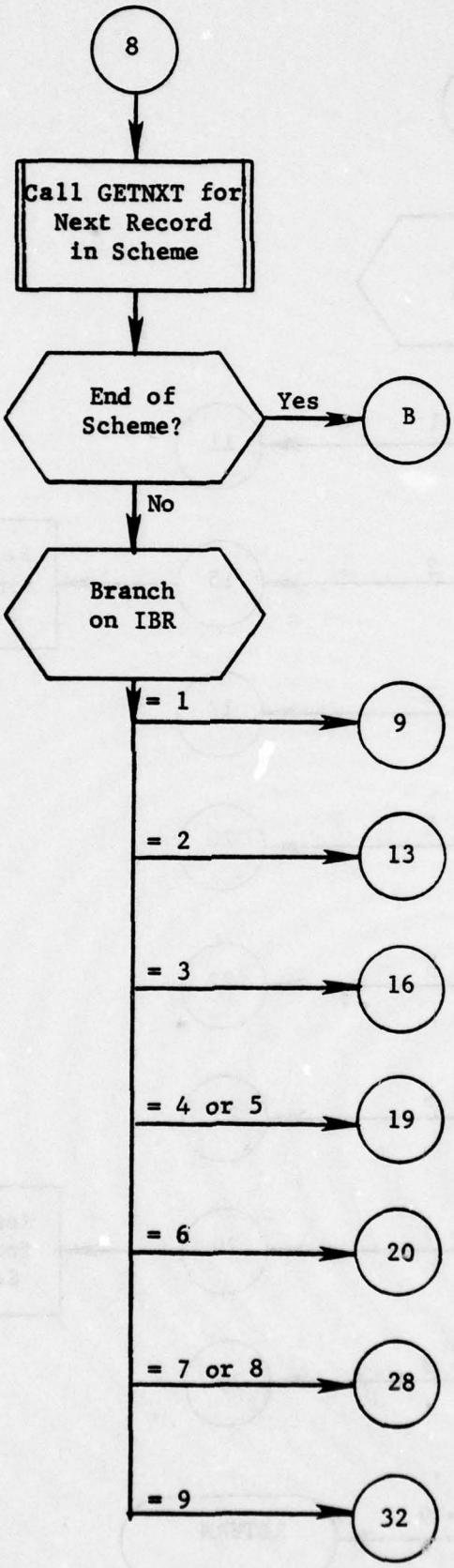


Figure 130. (Part 3 of 13)

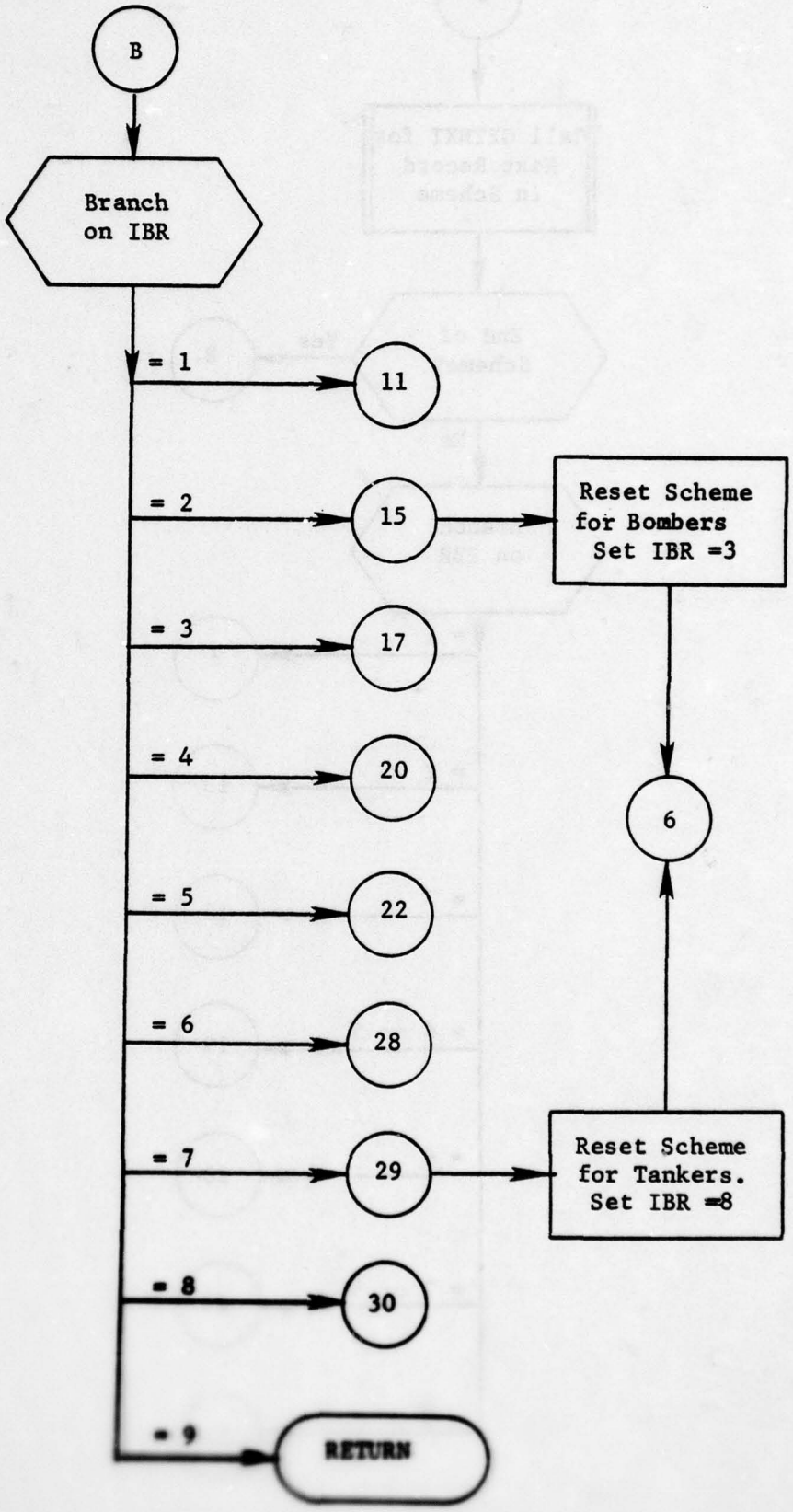


Figure 130. (Part 4 of 13)

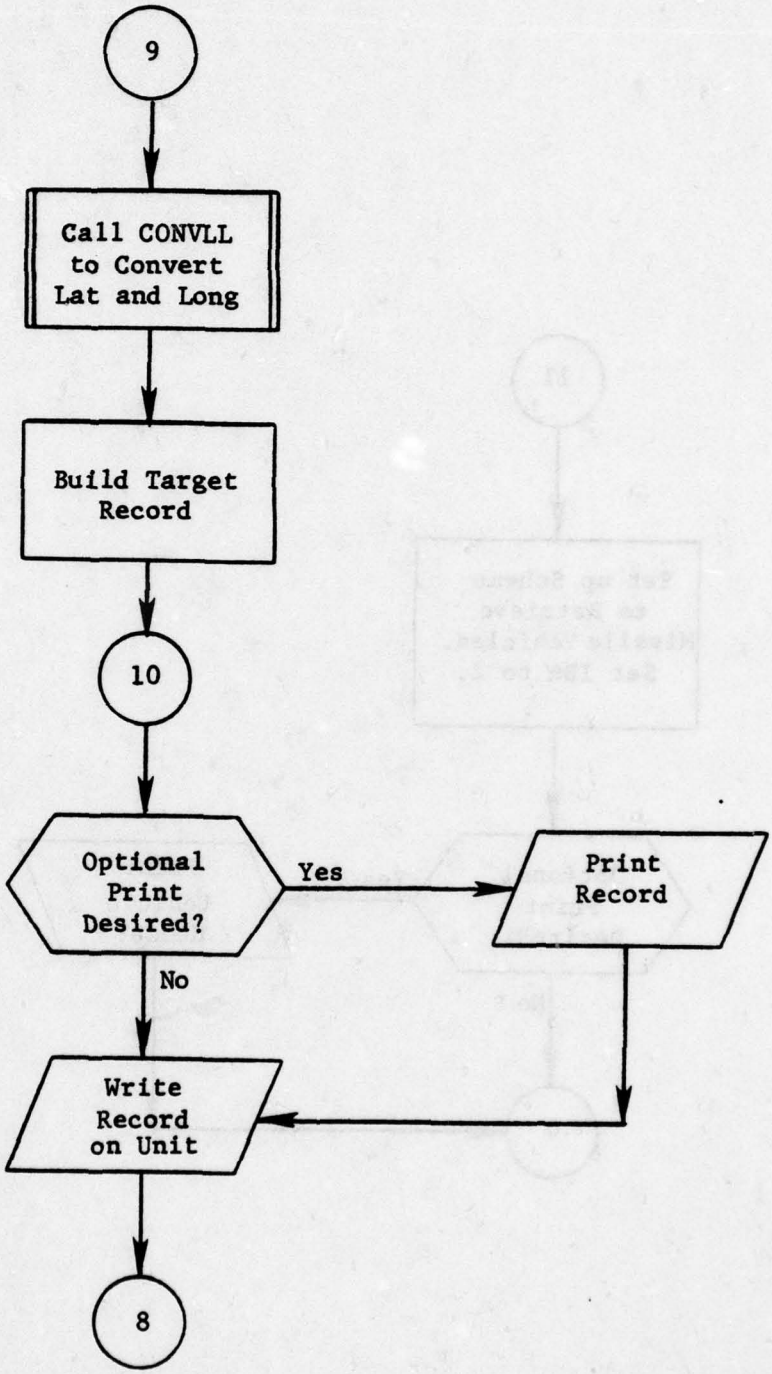


Figure 130. (Part 5 of 13)

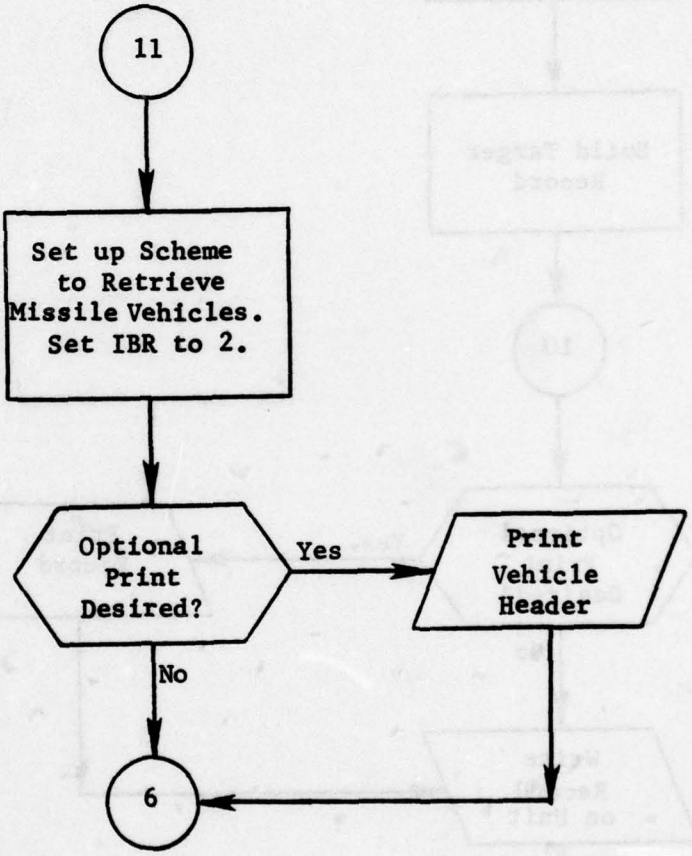


Figure 130. (Part 6 of 13)

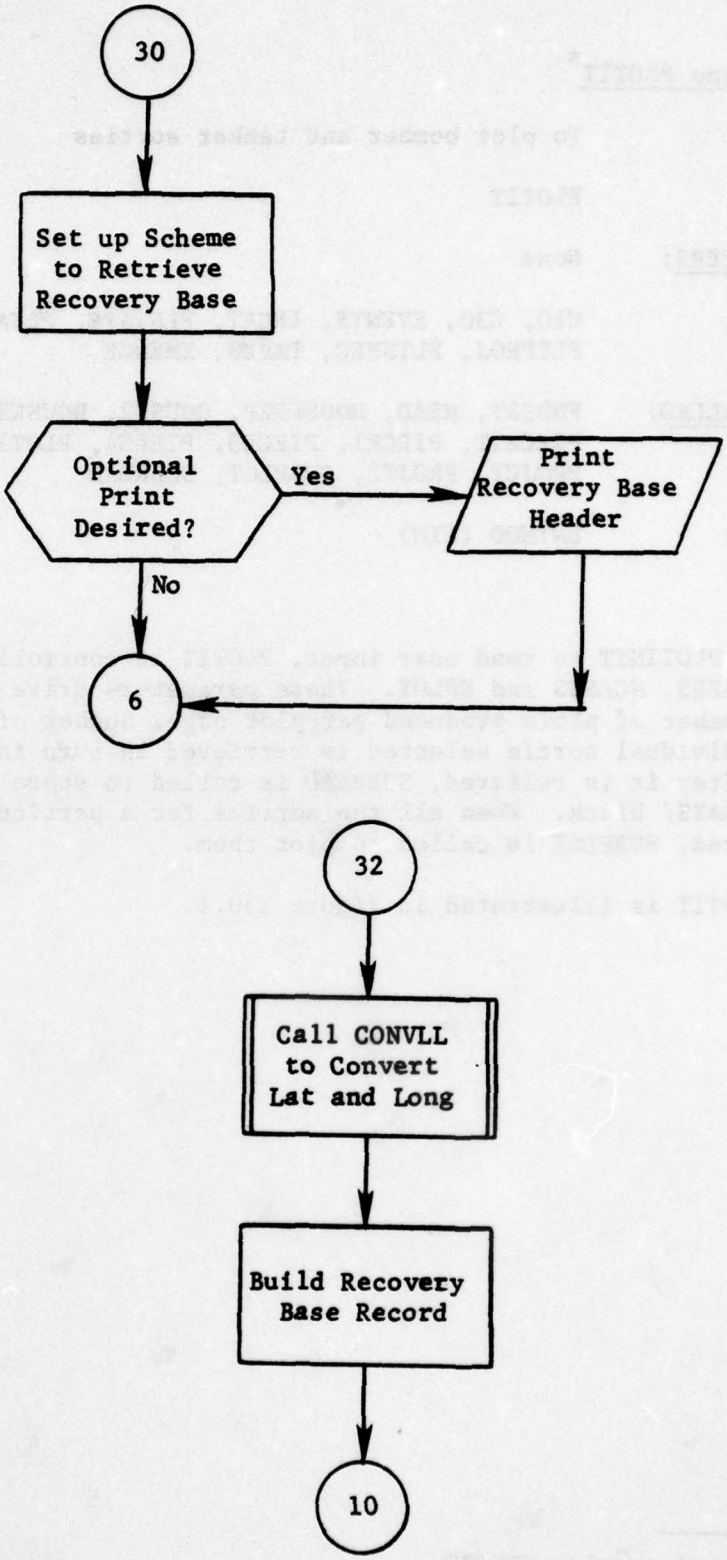


Figure 130. (Part 13 of 13)

8.12 Subroutine PLOTIT*

PURPOSE: To plot bomber and tanker sorties

ENTRY POINTS: PLOTIT

FORMAL PARAMETERS: None

COMMON BLOCKS: C10, C30, EVENTS, INCRT, PLNTAPE, PLTARRYS, PLTPROJ, PLTSPEC, TAPES, XMEDGE

SUBROUTINES CALLED: FNDSRT, HEAD, HOUSKEEP, HOUSK2, HOUSK3, NEXTTT, PIECEIT, PIECE1, PIECE3, PIECE4, PLOTINIT, PROJCT, PROJT2, SUBPLOT, SUBREAD

CALLED BY: ENTMOD (EIM)

Method:

After calling PLOTINIT to read user input, PLOTIT is controlled by the variables NGRAPHS, NCASES and NPLOT. These parameters drive loops that control the number of plots produced per plot page, number of pages, etc. Each individual sortie selected is retrieved in turn in the inner most loop. After it is relieved, SUBREAD is called to store the sortie in the /PLTARRAYS/ block. When all the sorties for a particular graph have been stored, SUBPLOT is called to plot them.

Subroutine PLOTIT is illustrated in figure 130.1.

* Main routine of overlay PLOTIT.

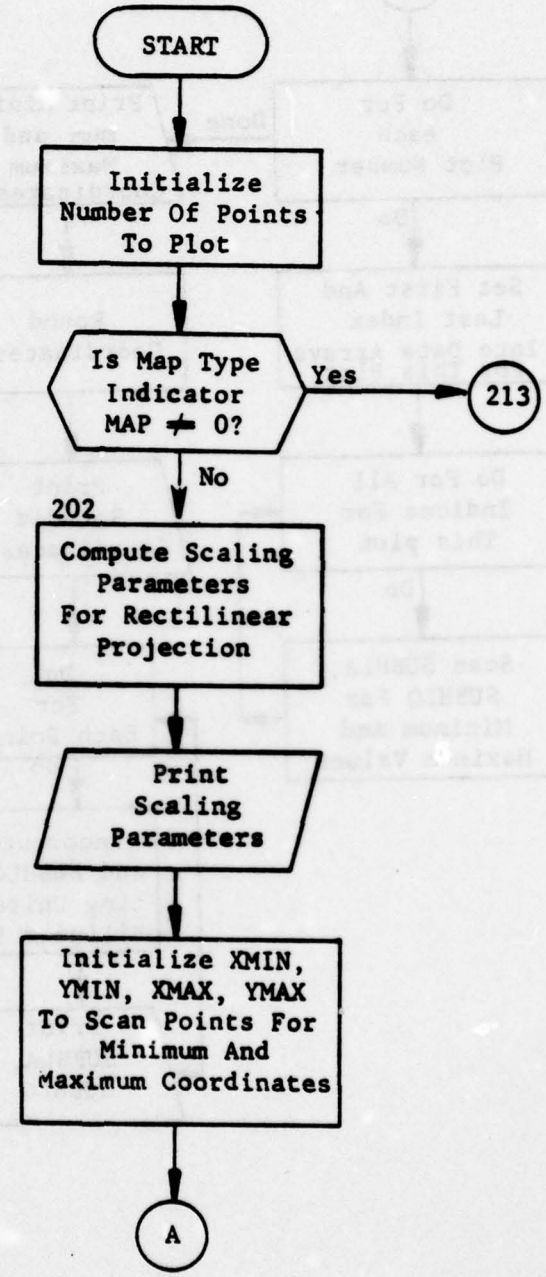


Figure 130.6. Subroutine SUBPLOT (Part 1 of 13)

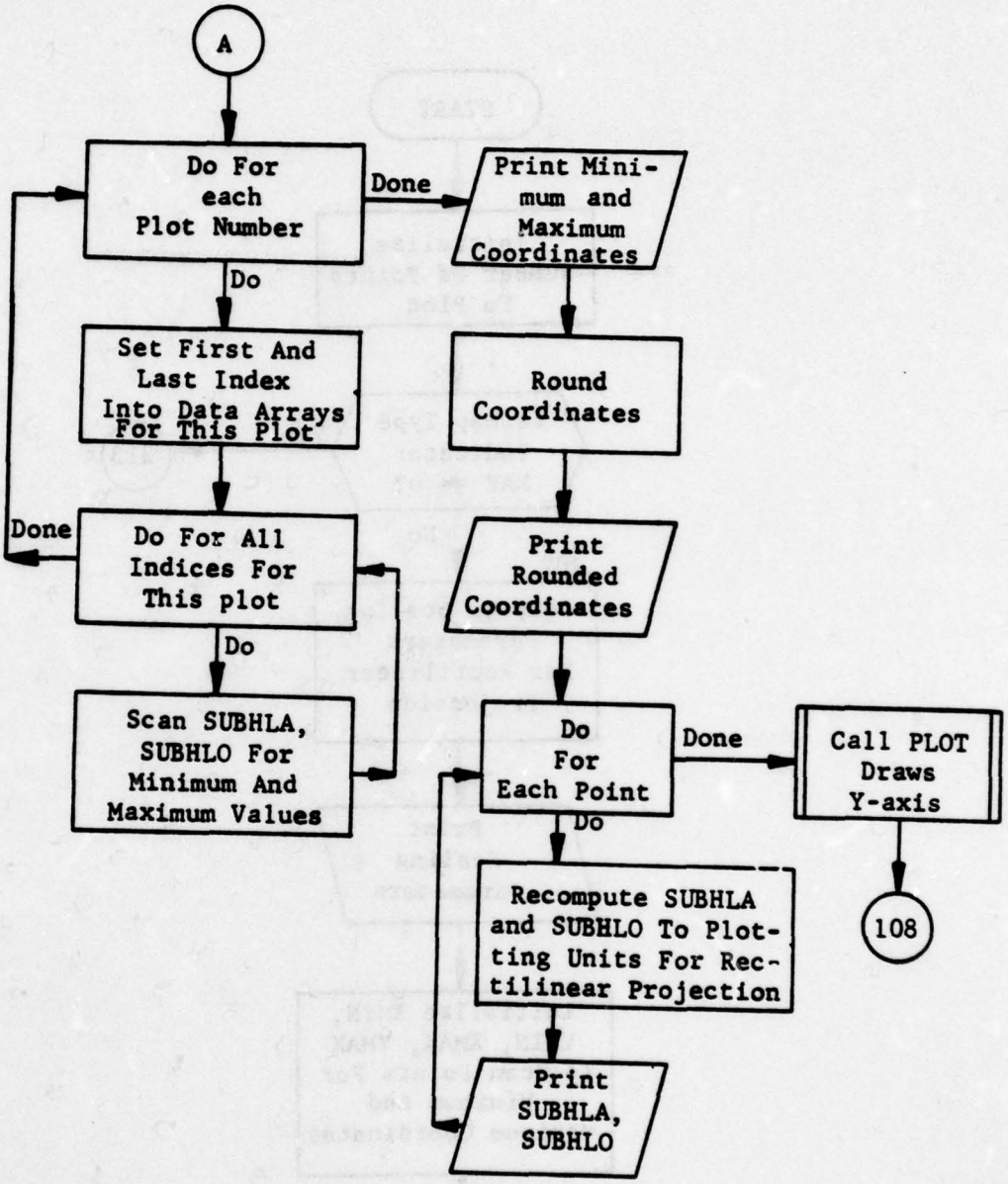


Figure 130.6. (Part 2 of 13)

Table 24. (Part 2 of 5)

Block	Array or Variable	Description
MISSLE	MSLDTA(50,3,6)	Missile time of flight data
	MRANGE(50)	Missile range
	NAMES(50)	Missile type names
ORDER	SCHORD(100)	Record type numbers in retrieval scheme order
	SORDNM(100)	Record type name in retrieval scheme order
	LENSCH	Length, in words, of retrieval scheme
PLTPROJ	MERCAT	Projection type
	IDIREC	Indicator of plot direction (+1 for counter - clockwise, -1 for clockwise)
	FLMDAO	Longitude of Origin
	PHIO	Latitude of Origin
	THETAO	Angle between meridian and x-axis
	PHI1	Standard parallel closest to equator
	PHI2	Standard parallel closest to pole
	A1L10	Fractional part of log for PHI1
	A2L10	Fractional part of log for PHI2
PLTSPEC	ISZE	Plot size indicator =0 for 50 x 40, =1 for 20 x 20, =2 for 10 x 10
	XAXLEN	Length of x-axis in inches
	YAXLEN	Length of y-axis in inches
	FACTOR	Number of plots per page

Table 24. (Part 3 of 5)

Block	Array or Variable	Description
	SCALE	Ratio of world unit to plot units
POLITE	S1	Latitude of beginning interpolation point
	T1	Longitude of beginning interpolation point
	S2	Latitude of interpolation end point
	T2	Longitude of interpolation end point
	FACTOR	Interpolation factor or fraction
	SR	Latitude of interpolated point
	TR	Longitude of interpolated point
PRNSP	PRINON	Switch to control optional prints, = True, produce print, = False, do not produce print
PSCOM		Used to communicate with subroutine PSREC
	RECORD(100)	Body of print/sort record
	RECLEN	Number of words in record
		Used to communicate with subroutine ATFNDR
	RTLIST(100)	List of record type numbers for retrieval scheme
	NUMREC	Number of record types in RTLIST
	HDREC	Record type name of primary header
RTLST	HCLASS	CLASS value of primary header
	HSIDE	SIDE value of primary header
	HOPT	CLASS/SIDE option for retrieval scheme

Table 24. (Part 4 of 5)

Block	Array or Variable	Description
	JHDR	Record type number of primary header
SCHEME	POINT	Pointer to current retrieval scheme instruction
	SCHEME(200)	Retrieval scheme (section 4.4)
SCRTCH	LIST(300)	Storage space used as work area by several subroutines
SIDES	SIDES(5)	List of values for SIDE
SNDMIN	XMIN	Minimum value of x-coordinator
	YATXMN	Y-coordinate at minimum X-coordinate
	XATYMN	X-coordinate at minimum Y-coordinate
	YMIN	Minimum value of y-coordinates
	ISUMIT	Number of points off the graph
SORSCH	SRTSCH(100)	Sort scheme (see section 6.10)
TAPES	PLOTTAPE	Logical unit number for plot tape
	PIECTAPE	Logical unit number for tape for non-plotted points
TARGET	SACB(1,2)	Target coordinates (Latitude and Longitude)
TSTUFF	TOFMIN	Minimum time of flight
	CMISS	Missile time coefficient
	RNGMIN	Minimum range
	RANGEM	Maximum range

Table 24. (Part 5 of 5)

Block	Array or Variable	Description
WAROUT	IWARFL	Logical unit number for printed output
XMEDGE	YMEDGE	Map edge
	XLL	X-coordinate of last point
	YLL	Y-coordinate of last point
	XL	X-coordinate of point to be plotted
	YL	Y-coordinate of point to be plotted
	XWEDGE	Converted value for latitude of origin
	BANGL	Converted value for longitude of origin

9.5 Function AZMUTH

PURPOSE:	Compute relative azimuth between two points on the earth for subroutine TOFM
ENTRY POINTS:	AZMUTH
FORMAL PARAMETERS:	RANGE - Surface range COSLT1 - Cosine of weapon latitude SINLT1 - Sine of weapon latitude SINLT2 - Sine of target latitude COSRNG - Cosine of great circle surface range
COMMON BLOCKS:	None
SUBROUTINES CALLED:	None
CALLED BY:	TOFM

Method:

The function computes missile flight azimuth according to the method described in "Program TRAJECT(u)-Foreigh Missile Flight Simulation Software - 29 Jan 79," 54474H, Aerospace Reconnaisance Technical Wing (SAC). Offutt AFC, Nebraska - Trajectory Division.

Function AZMUTH is illustrated in figure 134.

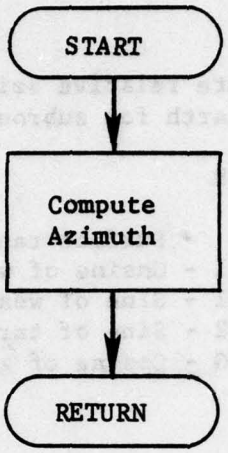


Figure 134. Function AZMUTH

9.6 Subroutine CINSGET

PURPOSE: Call INSGET and update index

ENTRY POINTS: CINSGET

FORMAL PARAMETERS:

L:	Array for data from INSGET call
LOC:	Starting index
N:	Number of words from INSGET call
LOCN:	Ending index +1

COMMON BLOCKS: None

SUBROUTINES CALLED: INSGET

Method:

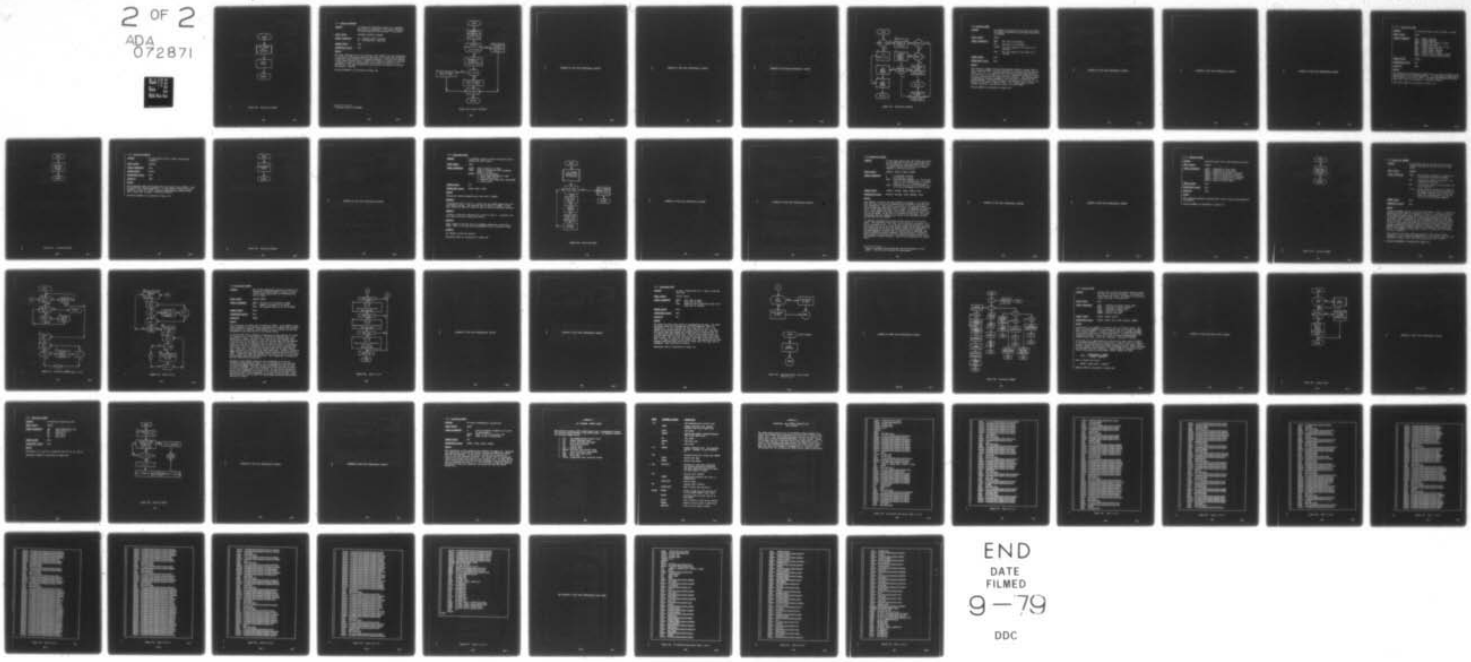
This subroutine calls INSGET in order to place N words into array L. LOC is the starting position into INSGETs arrays. LOCN is the next location into INSGET array's after CINSGET exits.

Subroutine CINSGET is illustrated in figure 135.

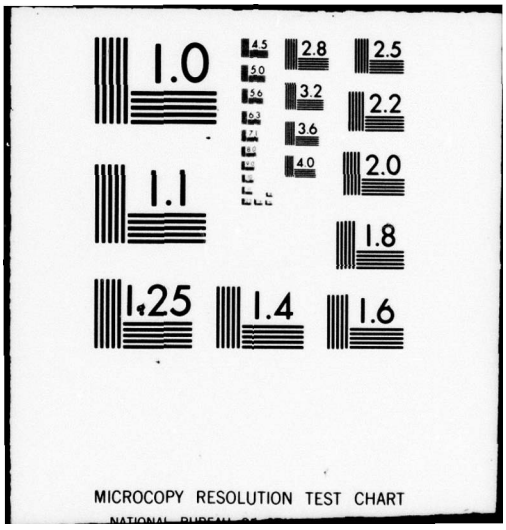
AD-A072 871 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON DC F/G 9/2
THE CCTC QUICK-REACTING GENERAL WAR GAMING SYSTEM (QUICK). PROG--ETC(U)
JUL 79

UNCLASSIFIED CCTC-CSM-MM-9-77-VI-CHG-2 NL

2 OF 2
ADA
072871



END
DATE
FILMED
9-79
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY RESOLUTION TEST CHART

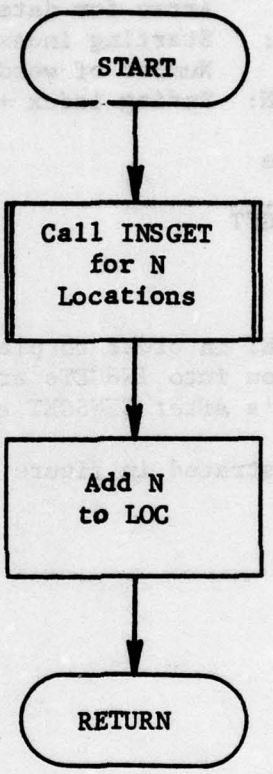


Figure 135. Subroutine CINSGET

9.7 Function DIFFLONG

PURPOSE: To compute the difference between two longitudes whose sign is determined by the shorter direction of travel from the first meridian to the second.

ENTRY POINTS: DIFFLONG, DIFFLNG*, DELLONG

FORMAL PARAMETERS: X1 - Floating point longitudes
X2 - Floating point longitudes

COMMON BLOCKS: None

SUBROUTINES CALLED: None

Method:

The input longitudes lie in the interval 0-360 degrees with west longitudes in the range 0-180 degrees and east longitudes in the range 180-360 degrees. Longitudes 0 and 360 define the Greenwich meridian. This function returns a value whose absolute value is equal to the number of degrees of longitude traversed in using the shorter great circle route from meridian X1 to meridian X2. The sign is positive if the direction of travel is eastward and negative otherwise.

Function DIFFLONG is illustrated in figure 136.

* Duplicate entry for DIFFLONG.

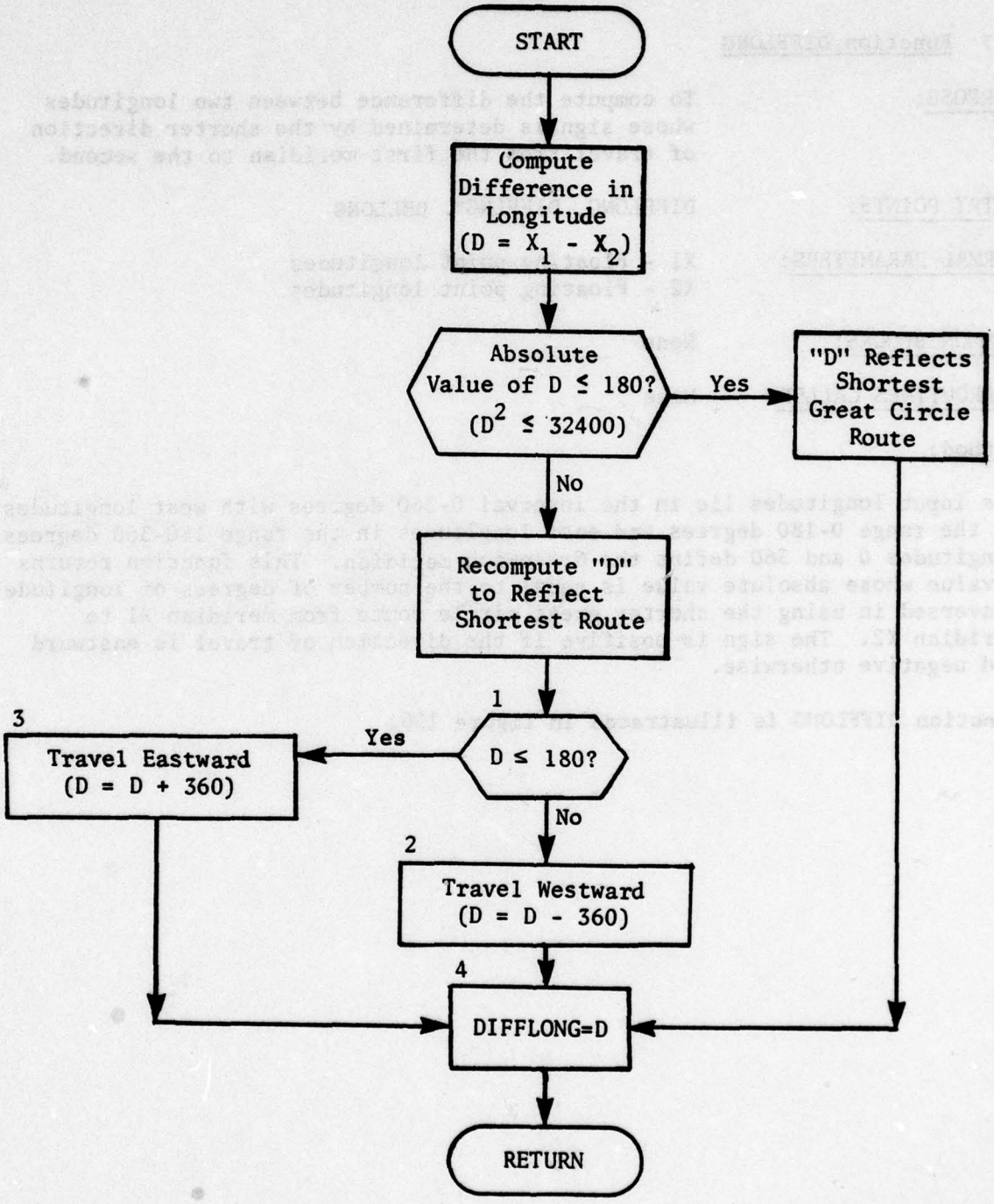
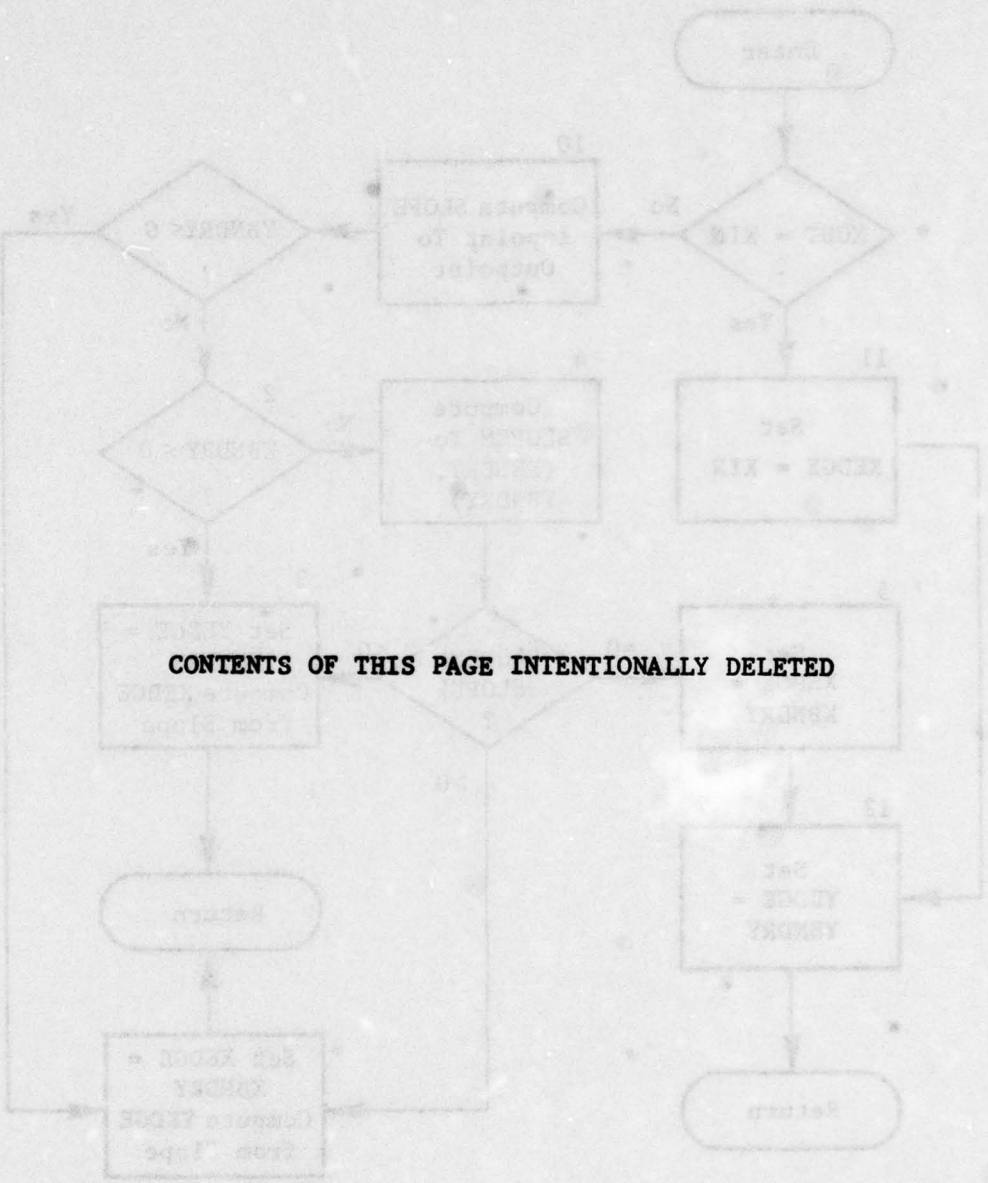


Figure 136. Function DIFFLONG

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

CONTENTS OF THIS PAGE INTENTIONALLY DELETED



CONTENTS OF THIS PAGE INTENTIONALLY DELETED

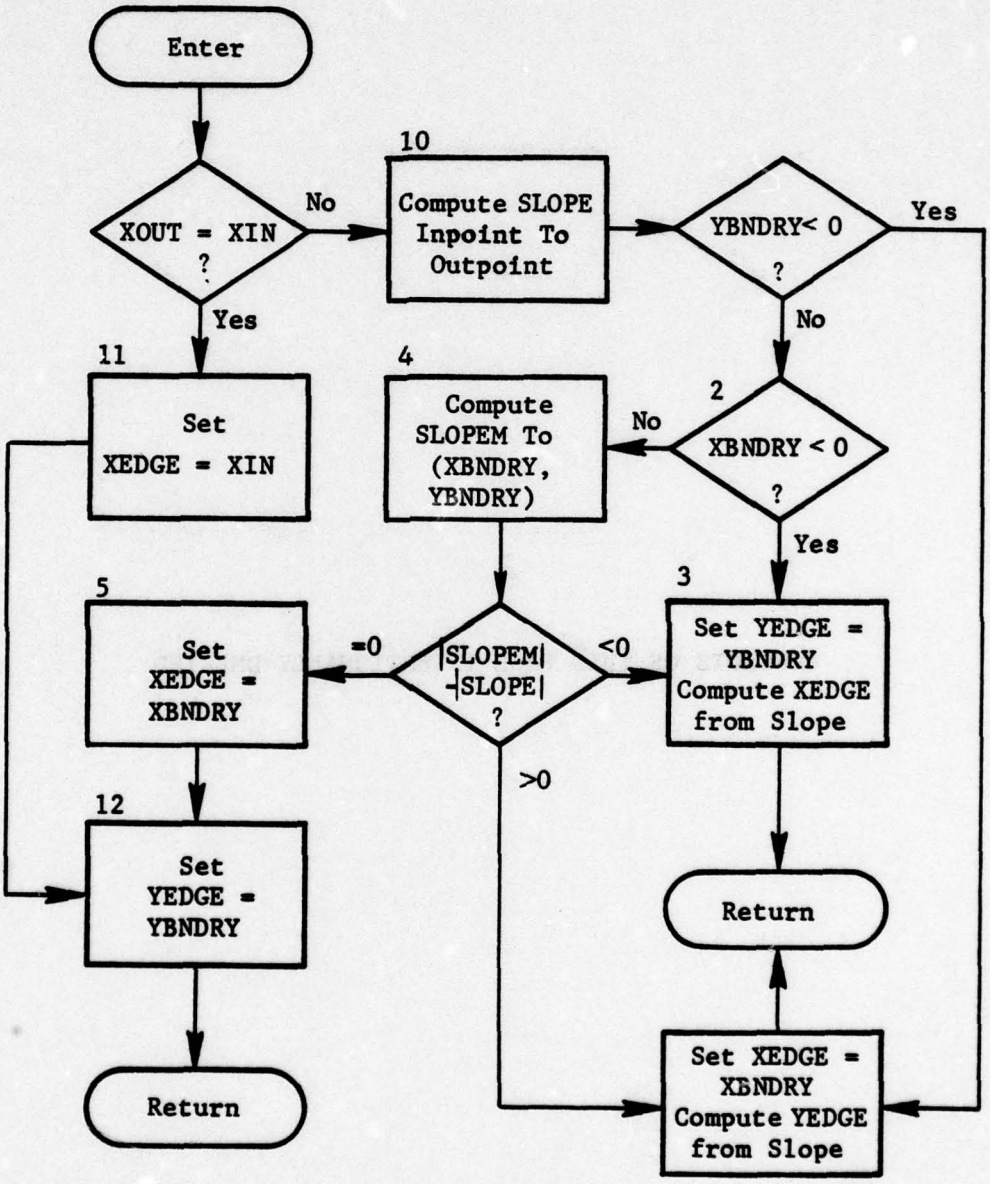


Figure 154. Subroutine INTPIECE

9.29 Function IWANT

PURPOSE: To determine the position of an item in an array. The search is modified by specifying items not to be searched.

ENTRY POINTS: IWANT

FORMAL PARAMETERS:

- NEED - The item to be matched
- IAR - The array to be searched
- ISTART - The first element in the array to be searched
- IEND - The last element in the array to be searched

COMMON BLOCKS: None

SUBROUTINES CALLED: None

Method:

This function searches array IAR from element ISTART to element IEND to find a match for NEED. At each occurrence of an element of IAR with the value = 1R*; i.e., 1-character BCD code for asterisk right-justified with zero fill to the left, the list is not searched for NEED until after the next occurrence of the right-justified asterisk. This search method corresponds to ignoring alphameric value subfields as defined by subroutine GETVALU. The value returned by the function is the index to the element of IAR which is equal to NEED. If no element meets the prescribed conditions, the function returns the value IEND+1

Function IWANT is illustrated in figure 159.

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

9.33.1 Subroutine LUNCH

PURPOSE: To calculate missile time-of-flight to target

ENTRY POINTS: LUNCH

FORMAL PARAMETERS:

LATL	- weapon latitude
LONL	- weapon longitude
AZNEW	- Azimuth from launch to target
MSL	- weapon type index
IRNG	- range index (min, mid, or max)
LATT	- output target latitude
LONT	- output target longitude
TOF	- output time of flight
COSLT	- cosine of output target latitude
SINLT	- sine of output target latitude

COMMON BLOCKS: MISSLE

SUBROUTINES CALLED: None

CALLED BY: TOFM

Method:

The subroutine computes missile flight time to a target according to the method described in "Program TRAJECT(u) - Foreign Missile Flight Simulation Software - 29 Jan 79," 5447H. Aerospace Reconnaissance Technical Wing (SAC), Offut AFT, Nebraska - Trajectory Division.

Subroutine LUNCH is illustrated in figure 163.1.

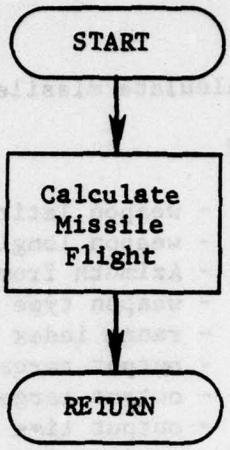


Figure 163.1. Subroutine LUNCH

9.35 Subroutine MISDATA

PURPOSE: To load missile time of flight initializing parameter

ENTRY POINTS: MISDATA

FORMAL PARAMETERS: None

COMMON BLOCKS: MISSLE

SUBROUTINES CALLED: None

CALLED BY: TOFM

Method:

This subroutine loads initializing data into common block /MISSLE/. Data is derived from "Program TRAJECT(u) - Foreign Missile Flight Simulation Software - 29 Jan 79," 5447H, Aerospace Reconnaissance Technical Wing (SAC), Offutt AFB, Nebraska - Trajectory Division.

Subroutine MISDATA is illustrated in figure 165.

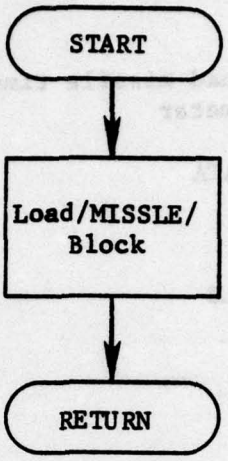


Figure 165. Subroutine MISDATA

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

9.36 Subroutine OFVAL

PURPOSE: To maintain a table of values retrieved from OF phrases and LIKE strings

ENTRY POINTS: OFVAL

FORMAL PARAMETERS:

VALUE: Value of phrase or string
INDEX: Index of instruction code containing phrase or string
BRANCH: Action to be performed
1 - Store VALUE and INDEX in table
2 - Flush table buffers
3 - Retrieve VALUE from table using INDEX
4 - Delete table

COMMON BLOCKS: C40

SUBROUTINES CALLED: DLETE, RETRV, STORE

Method:

The process differs depending upon the value of BRANCH.

BRANCH=1

If the buffer VALZ is full it is moved into the TABLEZ common block C40 and STORE is called. The reference code is saved in VLHDRZ. In any case, VALUE and INDEX are stored in VALZ in the next available position.

BRANCH=2

If there is only one table there is no need to save it. Otherwise, the last table is moved to TABLEZ and stored.

BRANCH=3

Every element of the old tables is examined looking for a match for INDEX. When it is found the corresponding value is returned in VALUE.

BRANCH=4

Any TABLEZs created are deleted.

Subroutine OFVAL is illustrated in figure 166.

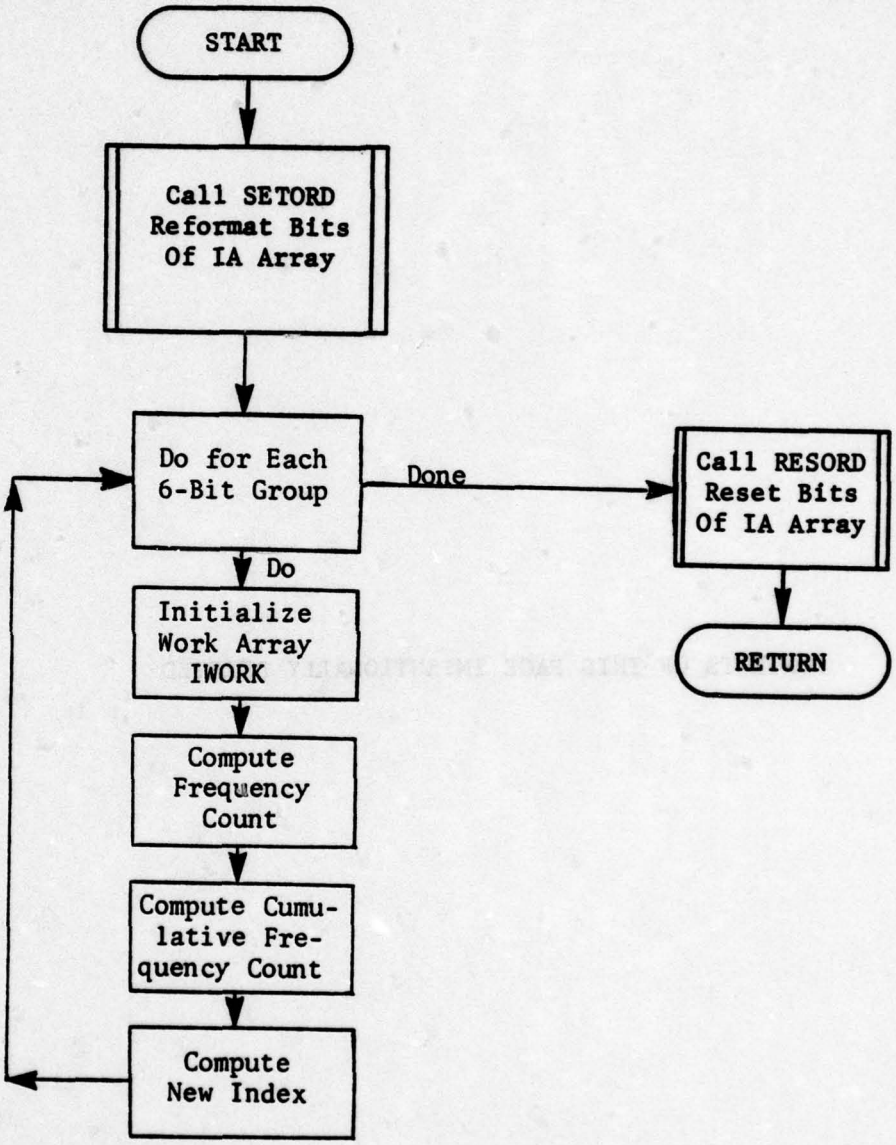


Figure 167. Subroutine ORDER

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

9.39 Subroutine PIECEIT

PURPOSE: To plot those points that fall within the limits of the graph and to write the points which are off the graph on a tape which may be used later to produce labeled plots which can be pieced together to form one large plot.

ENTRY POINTS: PIECEIT, PIECE1, PIECE3, PIECE4

FORMAL PARAMETERS:

- X - x-coordinate of point
- Y - y-coordinate of point
- MODE - Plotting mode indicator (= 1 for dotted line; = 2 for straight line; = 3 to move with pen up; = 4 to change pen color)
- IYESN - Indicator if point is on the graph (= 0 if it is on the graph; >0 if it is off)

COMMON BLOCKS: PLTPROJ, PLTSPEC, TAPES, SNDMIN, ISUP

SUBROUTINES CALLED: DOTLINE, INTPIECE, ISOFF, NEWPEN*, PLOT*

Method:

When PIECEIT is called, the mode indicator is tested. If it equals 4, only the color of the pen is changed and the PIECTAPE is written with that information. Then control is returned to the calling program. For any other mode subroutine ISOFF is called to determine if the point (X,Y) falls within the limits of the graph. If IYESN was returned with a value greater than zero, one or both coordinates of point (X,Y) are off the graph. No plotting is performed in this case. The point coordinates and mode indicator are output on the PIECTAPE and saved for the next call on PIECEIT.

If IYESN was returned with a value of zero, point (X,Y) is on the graph. The information saved from the previous point is tested if it was on the graph or not. If it was off, subroutine INTPIECE is called to position the pen at the point along the edge of the graph where the line connecting the previous and present points would intersect it. The mode indicator is tested and if a dotted line is requested, subroutine DOTLINE is called; otherwise, the PLOT routine is called to draw a line or move the pen from its present position to point (X,Y). The PIECTAPE is always written to keep all the information for further plots.

---

* These are standard plotting routines and not documented in this manual. They are only included for completeness.

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

9.44 Function RANGER

PURPOSE: Calculate great circle range between two points

ENTRY POINTS: RANGER

FORMAL PARAMETERS:

ALON1 - Longitude of first point
ALON2 - Longitude of second point
COSLT1 - Cosine of first point's latitude
SINLT1 - Sine of first point's latitude
COSLT2 - Cosine of second point's latitude
SINLT2 - Sine of second point's latitude
COSRNG - Cosine of output range

COMMON BLOCKS: None

SUBROUTINES CALLED: None

CALLED BY: TOFM

Method:

This function calculates standard great circle distance using spherical trigonometry.

Function RANGER is illustrated in figure 174.

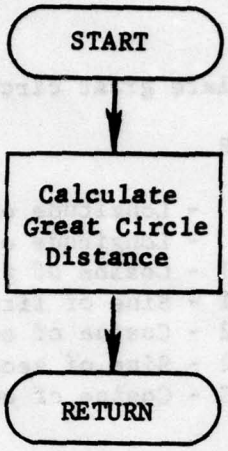


Figure 174.3. Function RANGER

9.45 Subroutine REORDER

PURPOSE: To rearrange from one to seven arrays into the sequence specified by the elements of an index array.

ENTRY POINTS: REORDER

FORMAL PARAMETERS:

- ISEQ - (Fixed point) sequence key array of the type produced by subroutine ORDER
- NEL - (Fixed point) number of elements to be reordered in each array
- NAR - (Fixed point) number of arrays which are to be reordered
- L1-L7 - (Fixed point or floating point) names of the arrays which are to be reordered; if the number of these arrays is less than seven, the remaining positions must be filled by trailing dummy arguments

COMMON BLOCKS: None

SUBROUTINES CALLED: None

Method:

Subroutine REORDER operates in the following manner. First, it stores one element from each array in a temporary location. It then reads from the array ISEQ the element which should go in that position (which may now be considered empty) and moves it, filling that position and creating a new blank. This is repeated for the corresponding element of each array being reordered. Each new blank spot is filled with its proper contents as soon as its original contents have been moved, until the element currently in temporary storage is required. When this happens, subroutine REORDER finds another element which is not already in its sequence and puts it into temporary storage and continues as before until no elements are out of sequence.

The contents of the array ISEQ are returned to the calling program unchanged, so that subroutine REORDER may be called again, using the same sequence key array if more than seven arrays are to be reordered.

Subroutine REORDER is illustrated in figure 175.

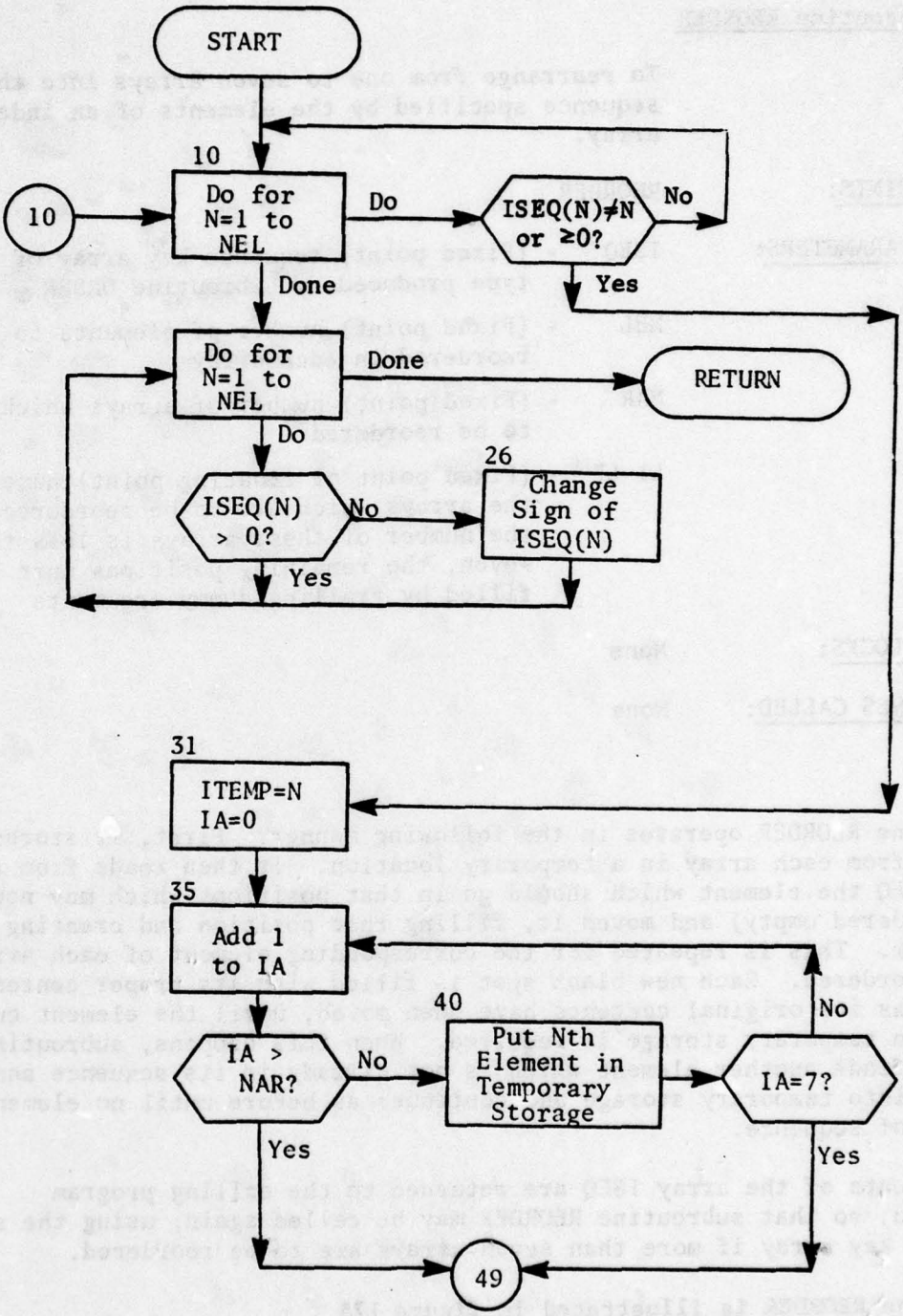


Figure 175. Subroutine REORDER (Part 1 of 2)

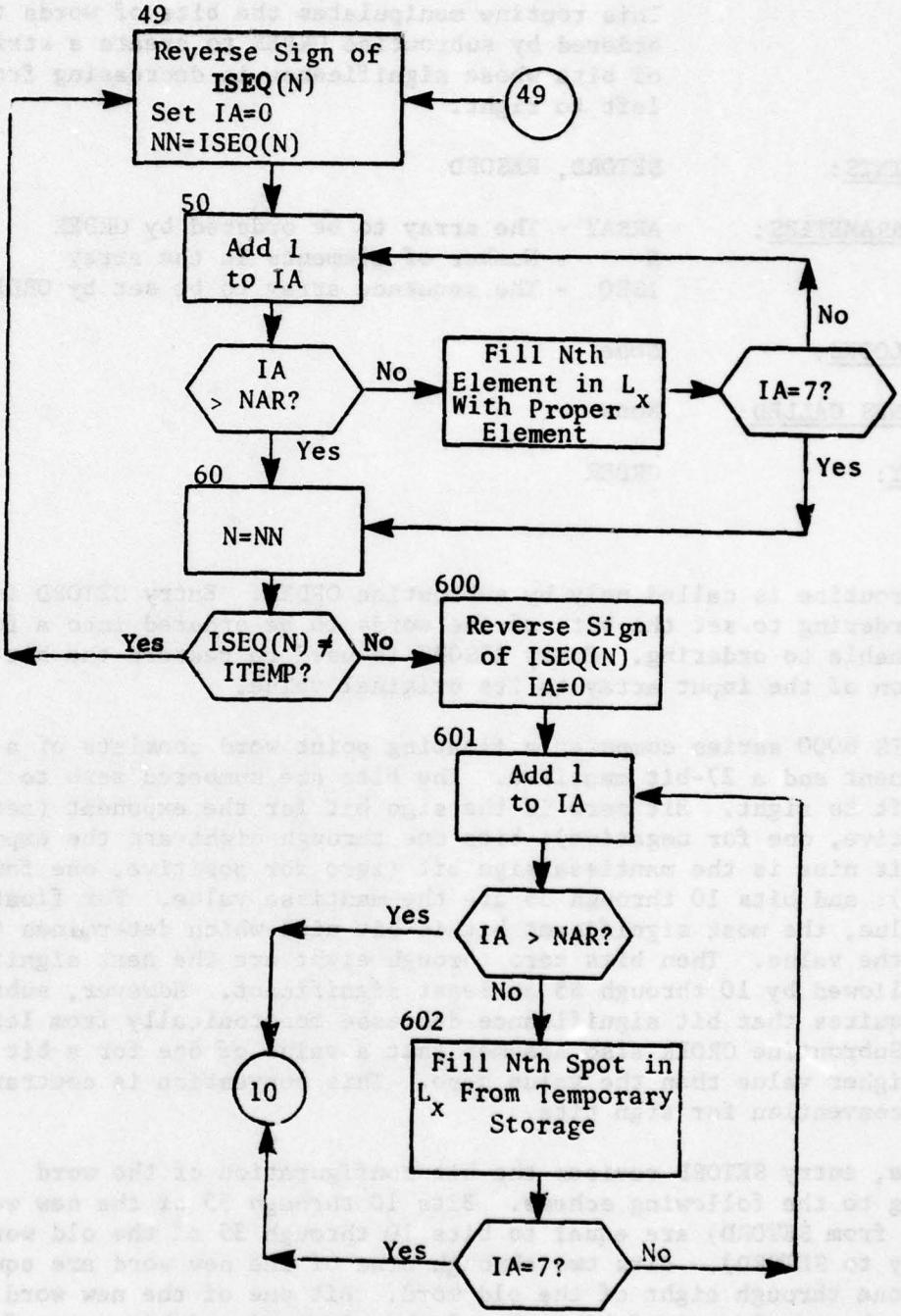


Figure 175. (Part 2 of 2)

9.46 Subroutine SETORD

PURPOSE: This routine manipulates the bits of words to be ordered by subroutine ORDER to create a string of bits whose significance is decreasing from left to right.

ENTRY POINTS: SETORD, RESORD

FORMAL PARAMETERS:
ARRAY - The array to be ordered by ORDER
N - Number of elements in the array
ISEQ - The sequence array to be set by ORDER

COMMON BLOCKS: None

SUBROUTINES CALLED: None

CALLED BY: ORDER

Method:

This subroutine is called only by subroutine ORDER. Entry SETORD is used before ordering to set the bits of the words to be ordered into a format more amenable to ordering. Entry RESORT is used to restore the bit configuration of the input array to its original value.

On the HIS 6000 series computer a floating point word consists of a nine-bit exponent and a 27-bit mantissa. The bits are numbered zero to 35 going left to right. Bit zero is the sign bit for the exponent (zero for positive, one for negative); bits one through eight are the exponent value; bit nine is the mantissa sign bit (zero for positive, one for negative); and bits 10 through 35 are the mantissa value. For floating point value, the most significant bit is bit nine which determines the sign of the value. Then bits zero through eight are the next significant, followed by 10 through 35 as least significant. However, subroutine ORDER requires that bit significance decrease monotonically from left to right. Subroutine ORDER also assumes that a value of one for a bit signifies a higher value than the value zero. This convention is contrary to the HIS convention for sign bits.

Therefore, entry SETORD revises the bit configuration of the word according to the following scheme. Bits 10 through 35 of the new word (at exit from SETORD) are equal to bits 10 through 35 of the old word (at entry to SETORD). Bits two through nine of the new word are equal to bits one through eight of the old word. Bit one of the new word is equal to the complement of bit zero of the old word and bit zero of the new word is equal to the complement of bit nine of the old word. This scheme produces a word in which bit significance decreases monotonically from left to right.

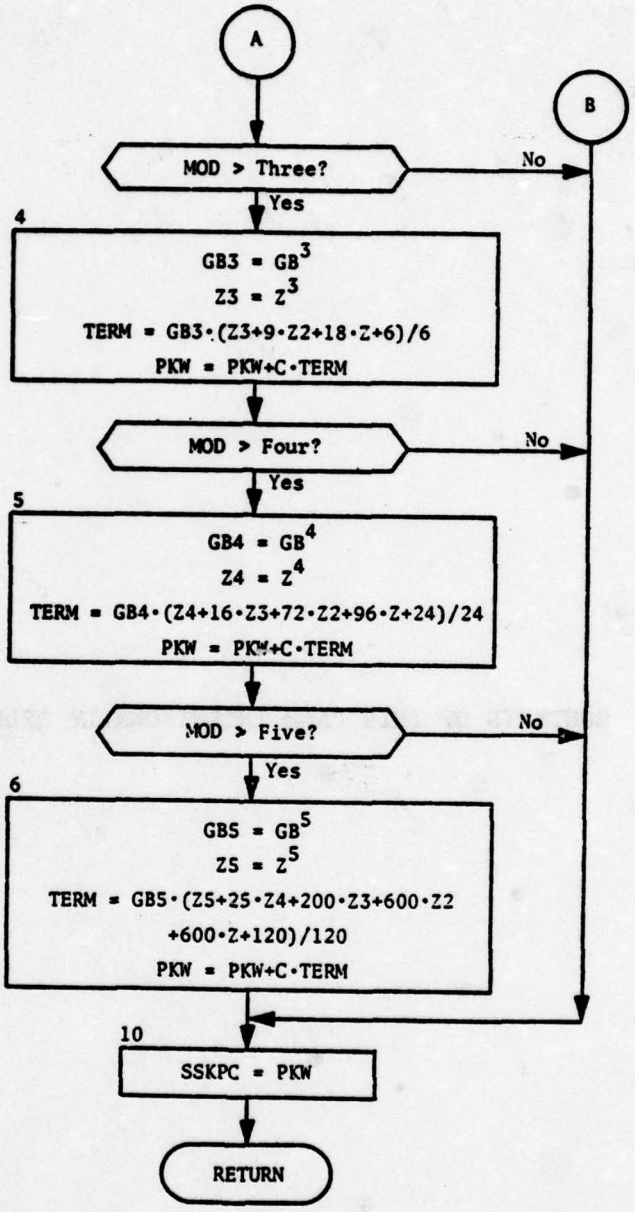
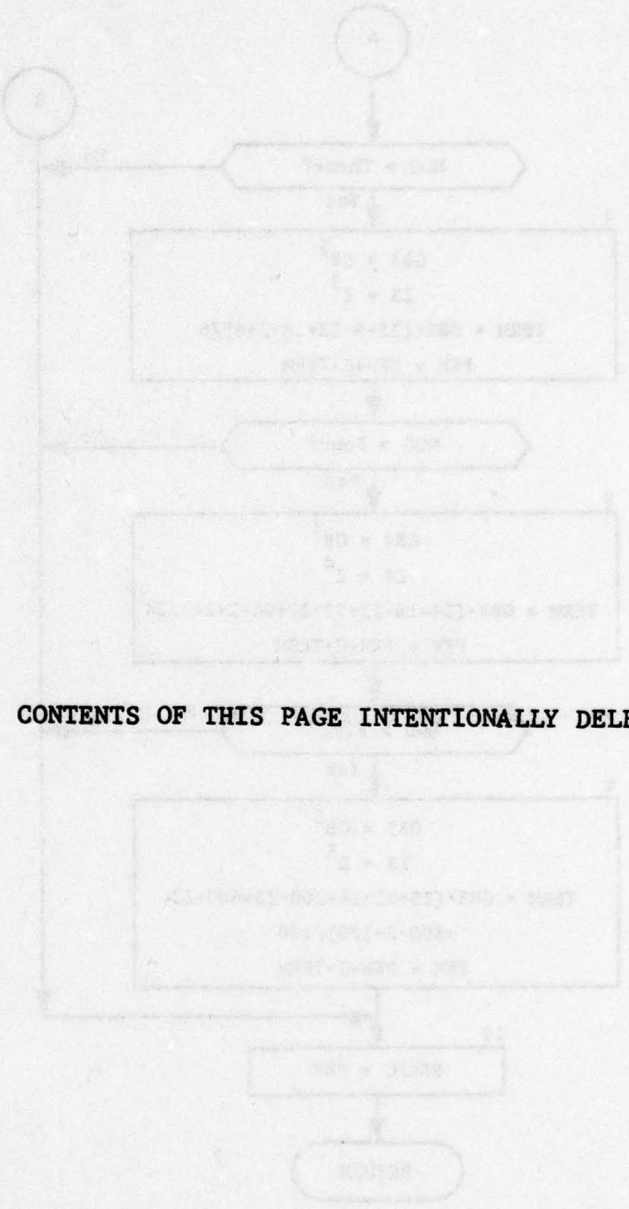


Figure 180. (Part 2 of 2)



CONTENTS OF THIS PAGE INTENTIONALLY DELETED

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

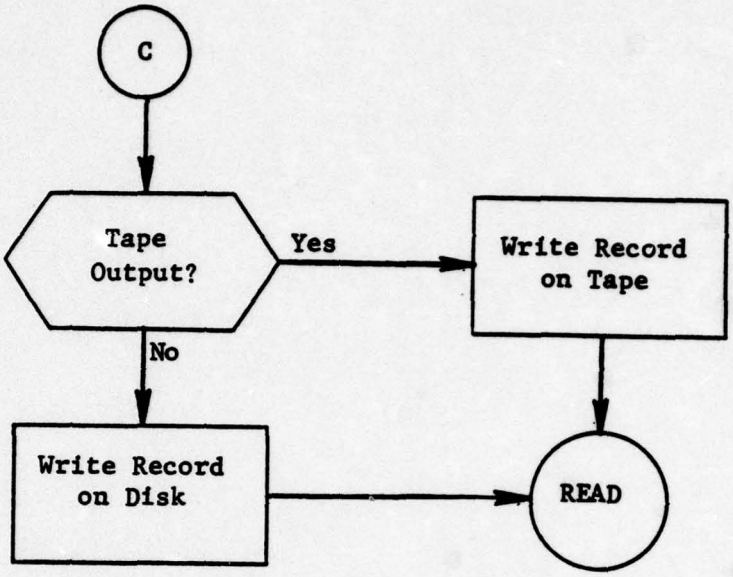
9.52 Subroutine SVTP

PURPOSE:	To save a random disk file on tape or load one from tape
ENTRY POINTS:	FILLOD, FILSAV
FORMAL PARAMETERS:	FROM: File code of input TO: File code of output FILE: Number of file on multifile tape to be used (may be omitted)
COMMON BLOCKS:	None
SUBROUTINES CALLED:	None
CALLED BY:	SRM

Method:

The method for both entry points is approximately the same. The input and output codes are retrieved and the third parameter (FILE) is checked for. If it is omitted the first tape file is used. A 4k buffer is now created and the file control blocks set up. The input and output codes are checked for validity and it is assured that one is tape and the other random disk. Here the entry points differ in that the input unit to FILLOD must be tape and the output from FILSAV must be tape. The files are now opened, depending on the third parameter, the tape is spaced to the proper file. The subroutine now loops through reads from input and writes to output until the input unit is exhausted. Then the files are closed.

Subroutine SVTP is illustrated in figure 182.



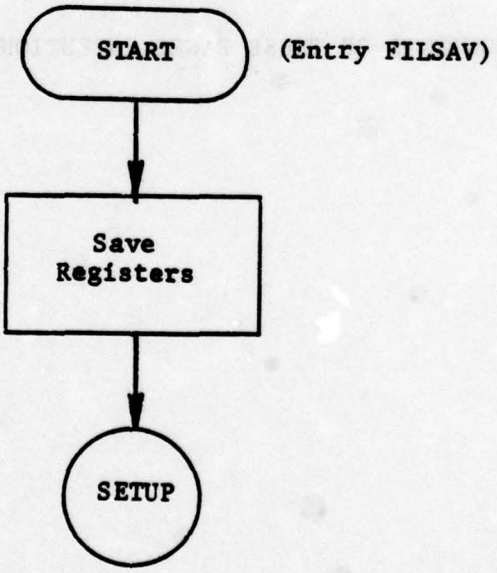
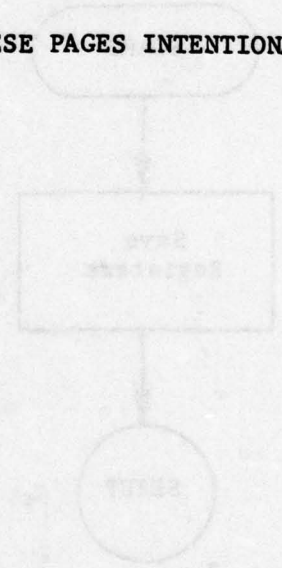


Figure 182. Subroutine SVTP: Entry FILSAV
(Part 5 of 5)

CONTENTS OF THESE PAGES INTENTIONALLY DELETED



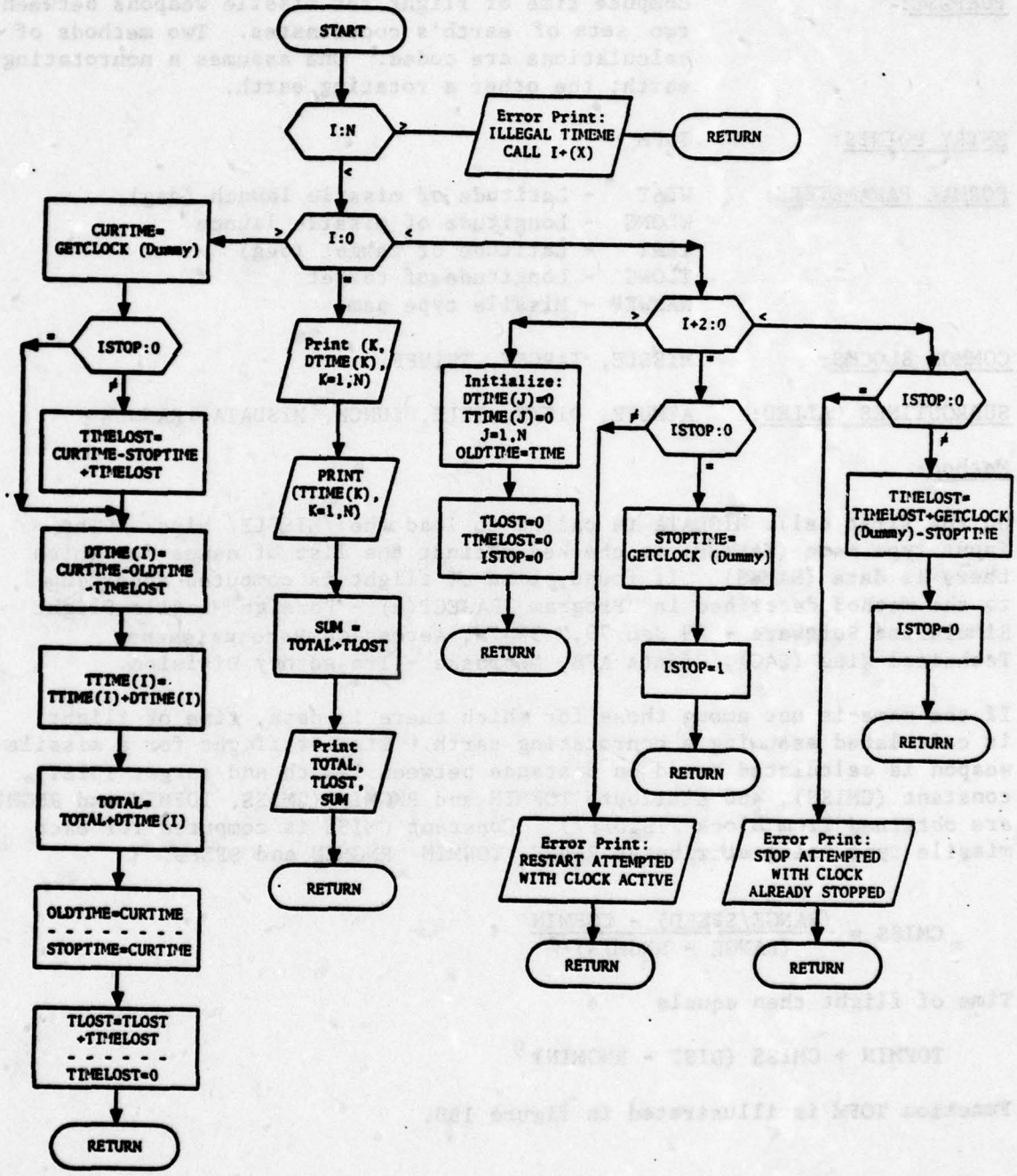


Figure 185. Subroutine TIMEME

9.56 Function TOFM

PURPOSE: Compute time of flight for missile weapons between two sets of earth's coordinates. Two methods of calculations are coded. One assumes a nonrotating earth; the other a rotating earth.

ENTRY POINTS: TOFM

FORMAL PARAMETERS:
- WLAT - Latitude of missile launch (deg)
- WLONG - Longitude of missile launch
- TLAT - Latitude of target (deg)
- TLONG - Longitude of target
- NAMWEP - Missile type name

COMMON BLOCKS: MISSLE, TARGET, TSTUFF

SUBROUTINES CALLED: AZMUTH, DISTF, ITLE, LUNCH, MISDATA, RANGER

Method:

On the first call, MISDATA is called to load the /MISSLE/ block. The input type name (NAMWEP) is checked against the list of names for which there is data (NAMES). If found, time of flight is computed according to the method described in "Program TRAJECT(u) - Foreign Missile Flight Simulation Software - 29 Jan 79," 5447H, Aerospace Reconnaissance Technical Wing (SAC), Offutt AFB, Nebraska - Trajectory Division.

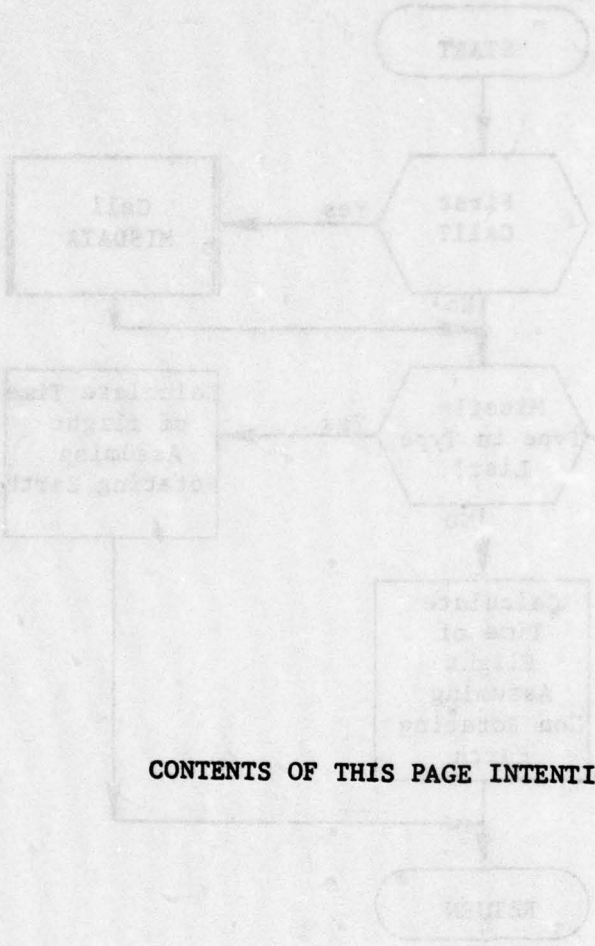
If the name is not among those for which there is data, time of flight is calculated assuming a nonrotating earth. Time of flight for a missile weapon is calculated based on distance between launch and target (DIST), constant (CMISS), and attribute TOFMIN and RNGMIN (CMISS, TOFMIN and RNGMIN are obtained from block /TSTUFF/). Constant CMISS is computed for each missile type using attributes RANGE, TOFMIN RNGMIN and SPEED.

$$CMISS = \frac{(RANGE/SPEED) - TOFMIN}{(RANGE - RNGMIN)^{.9}}$$

Time of flight then equals

$$TOFMIN + CMISS\ (DIST - RNGMIN)^{.9}$$

Function TOFM is illustrated in figure 186.



CONTENTS OF THIS PAGE INTENTIONALLY DELETED

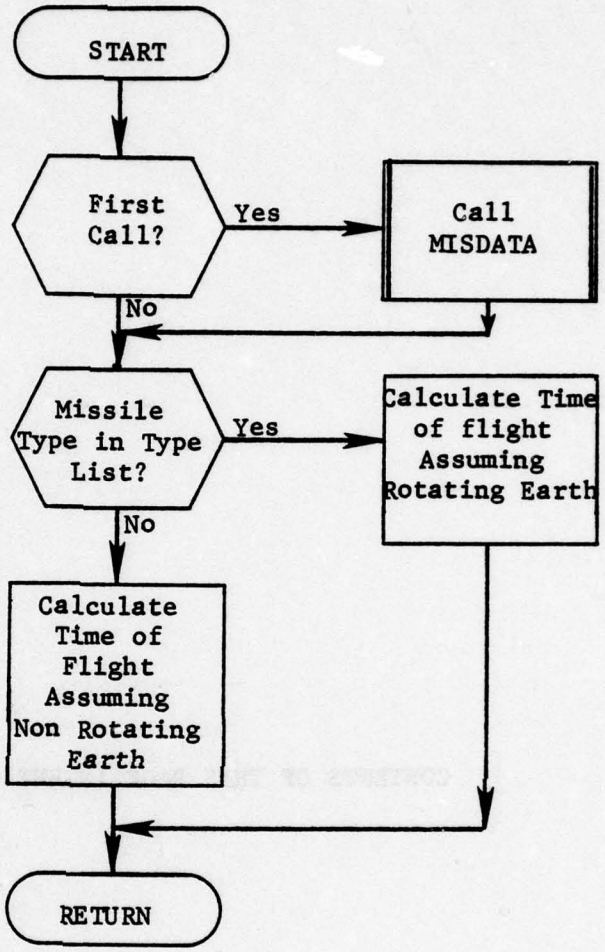


Figure 186. Function TOFM

CONTENTS OF THESE PAGES INTENTIONALLY DELETED

9.57 Subroutine UNCODE

PURPOSE: To decode an instruction code

ENTRY POINTS: UNCODE

FORMAL PARAMETERS:

INP:	Input instruction code
IW:	(bits 29-32) +1
IX:	bits 33-35
IY:	(bit 33) +1
IZ:	bits 34-35

COMMON BLOCKS: None

SUBROUTINES CALLED: None

Method:

The method is to use FLD to break INP into IW, IX, IY, and IZ.

Subroutine UNCODE is illustrated in figure 187.

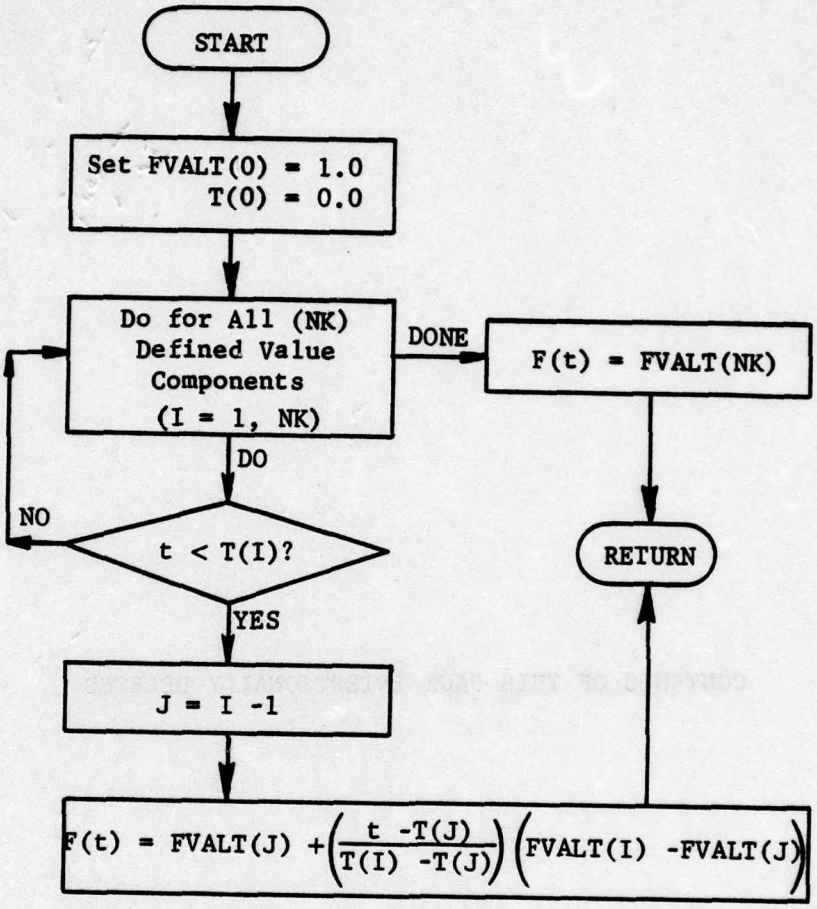
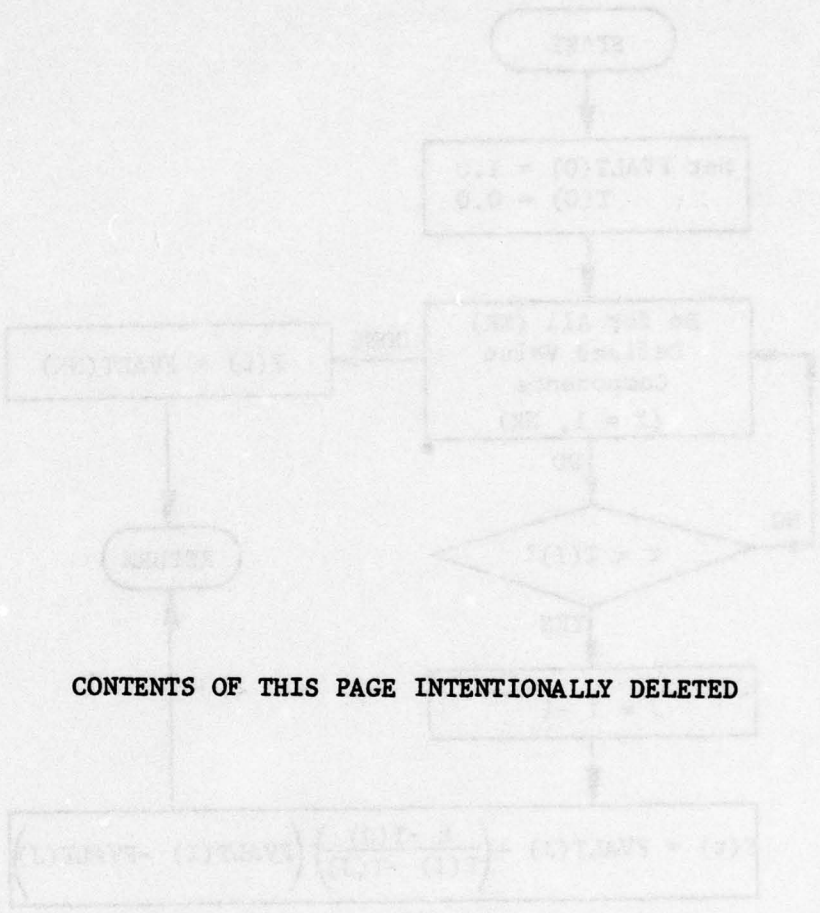


Figure 189. Function VALTAR



CONTENTS OF THIS PAGE INTENTIONALLY DELETED

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

9.61 Subroutine XMATH

PURPOSE: To execute mathematical calculations

ENTRY POINTS: XMATH

FORMAL PARAMETERS:

X:	Array of internal variables to be used in calculations
BEGIN:	Index of first instruction code
END:	Index of end of calculation

COMMON BLOCKS: C30

SUBROUTINES CALLED: INSGET, IORFL, OFVAL, UNCODE

Method:

This subroutine is best understood by reference to figure 191. Basically the instructions retrieved from INSGET starting with BEGIN and ending with END are executed. In the process, values will be stored in X which is used for internal variables. The current value is maintained locally in Q and the new value is stored locally in R. For each instruction a branch (IBR) is set and then the remainder of the instruction is used to determine the value of R. Then the branch is made and whatever operation on Q and R is called for is carried out with the result placed in Q.

APPENDIX A

COP EXTERNAL COMMON BLOCKS

This appendix contains those common blocks used to communicate between the COP and related modules of the QUICK system. The appendix contains the following common blocks:

- o C10 IDS Communications control block
- o C15 Header reference codes
- o C20 Record type name and number
- o C30 Data base attributes
- o C40 Utility table
- o C50 Display table
- o ERRCOM IDS error code control block
- o INS Input instruction code buffer
- o IPGT Input card image buffer
- o OOPS Error flag
- o STRING Interpreted input character string

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
C10		IDS communications control block
	IREFZ	Binary reference code, updated whenever IDS action takes place
	MQ(2)	(Not used)
	IRECTP	Record type number, updated whenever IDS action takes place
	NQ	(Not used)
	ERCODE	IDS error code
	NQ2	(Not used)
C15	HEADRF	Header reference code. BCD representation. Variable is type character *8
C20		Contains values for record type INDRCT
	RNAME	Record type name
	RNUMB	Record type number
C30	MAIN(306)	Contains all data base attributes. For precise attributes definition and their addresses within the array see Users Manual, Volume I
C40		Utility table (TABLEZ)
	TABREF	TABLEZ BCD reference code (type is character*8)
	TABLE(100)	Body of table
C50		Display table (DISPDT)
	DSPTAB(100)	Body of table (see section 6)
ERRCOM	NORMAC	Action to take if error code is not in list CHEKS (ABORT, FLAG, PASS)
	CHCKAC	Action to take if error code is in list CHEKS
	ERUNIT	Unit on which to print error message
	NUMCHK	Number of error codes in CHEKS list
	CHEKS(30)	List of error codes to check

APPENDIX B

EXECUTABLE JOB CONTROL LANGUAGE (JCL)
QUICK SYSTEM

The QUICK system executes from an H*(HIS6000 System Loadable file). Figure 194 contains the JCL necessary to build the H* directly from source files. Note that within the FORTY activity for each link an object file (catalog 632IDP00/QUIK/OBJECT) is also created. Figure 195 contains the JCL to recreate the H* from these object files. In effect, the programmer needs only replace the SELECT of the object deck with the corresponding FORTY activity for each overlay link in which the source has changed to recreate the H*. Finally, figure 196 contains the JCL to create the QUICK utility library from source.

```
$       IDENT   1820510/30/5162,COMPL
$       USERID  632IDP99$PASSWORD/UZZ
$       LIMITS  30,60K,,50K
$       OPTION  FORTRAN,NOGO
$       LIBRARY UL,PL
$       LOWLOAD
$       ENTRY   ......
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/COP
$       SELECT  632IDP00/QUIK/SOURCE/COP/COP
$       SELECT  632IDP00/QUIK/SOURCE/COP/BANNER
$       SELECT  632IDP00/QUIK/SOURCE/COP/ERPROC
$       SELECT  632IDP00/QUIK/SOURCE/COP/HDFND
$       SELECT  632IDP00/QUIK/SOURCE/COP/INICOP
$       SELECT  632IDP00/QUIK/SOURCE/COP/INPRIN
$       SELECT  632IDP00/QUIK/SOURCE /COP/ INSPUT
$       SELECT  632IDP00/QUIK/SOURCE/COP/MODGET
$       IDS     DECK
$       LIMITS  30,60K,,50K
$       FILE    *3,X3R,100R
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/QDATA
$       SELECT  632IDP00/QUIK/SOURCE/COP/QDATA
$       SELECT  632IDP00/QUIK/SOURCE/COP/QDATB
$       USE     .QMAX/1/,.QAREA/3126/,.QMIN/1/,.FRRD.
$       LINK    BOOTT
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/BOOT
$       SELECT  632IDP00/QUIK/SOURCE/BOOT
$       SELECT  632IDP00/QUIK/SOURCE/COP/DCTFND
$       SELECT  632IDP00/QUIK/SOURCE/COP/MNMFND
$       SELECT  632IDP00/QUIK/SOURCE/COP/NUMFND
$       SELECT  632IDP00/QUIK/SOURCE/COP/RNMFND
$       SELECT  632IDP00/QUIK/SOURCE/COP/SEEKER
$       SELECT  632IDP00/QUIK/SOURCE/COP/STRMAK
$       LINK    TABSTR,BOOTT
$       USE     TABLZ/808/
$       LINK    ERRF
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/ERRFND
$       SELECT  632IDP00/QUIK/SOURCE/COP/ERRFND
$       SELECT  632IDP00/QUIK/SOURCE/COP/LNGSTR
$       SELECT  632IDP00/QUIK/SOURCE/COP/SYNTAX
$       SELECT  632IDP00/QUIK/SOURCE/COP/TABINS
$       SELECT  632IDP00/QUIK/SOURCE/COP/WEBSTR
$       LINK    INPT,ERRF
$       FORTY   MAP,XREF,DECK
```

Figure 194. H* Creation From Source (Part 1 of 11)

```
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/INPTRN
$       SELECT  632IDP00/QUIK/SOURCE/COP/INPTRN
$       SELECT  632IDP00/QUIK/SOURCE/COP/DELTAB
$       SELECT  632IDP00/QUIK/SOURCE/COP/INMATH
$       SELECT  632IDP00/QUIK/SOURCE/COP/LINEIO
$       SELECT  632IDP00/QUIK/SOURCE/COP/PARLEV
$       SELECT  632IDP00/QUIK/SOURCE/COP/TABGET
$       LINK    JLM,TABSTR
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/JLM
$       SELECT  632IDP00/QUIK/SOURCE/JLM/JLM
$       LINK    ASSI
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/ASSIGN
$       SELECT  632IDP00/QUIK/SOURCE/JLM/ASSIGN
$       SELECT  632IDP00/QUIK/SOURCE/JLM/ALPHAS
$       SELECT  632IDP00/QUIK/SOURCE/JLM/PLAYERS
$       SELECT  632IDP00/QUIK/SOURCE/JLM/TOPRINT
$       LINK    SELE,ASSI
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/SELECT
$       SELECT  632IDP00/QUIK/SOURCE/JLM/SELECT
$       SELECT  632IDP00/QUIK/SOURCE/JLM/ADTOBASE
$       SELECT  632IDP00/QUIK/SOURCE/JLM/DEFAULT
$       SELECT  632IDP00/QUIK/SOURCE/JLM/KRUNCH
$       SELECT  632IDP00/QUIK/SOURCE/JLM/SAMSET
$       LINK    ASTE,SELE
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/ASTERISK
$       SELECT  632IDP00/QUIK/SOURCE/JLM/ASTERISK
$       LINK    DATA,JLM
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/DATA
$       SELECT  632IDP00/QUIK/SOURCE/DATA/DATA
$       LINK    DATADL
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/DELETE
$       SELECT  632IDP00/QUIK/SOURCE/DATA/DELETE
$       LINK    DATACH,DATADL
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/CHANGE
$       SELECT  632IDP00/QUIK/SOURCE/DATA/CHANGE
$       SELECT  632IDP00/QUIK/SOURCE/DATA/DESSCH
$       SELECT  632IDP00/QUIK/SOURCE/DATA/NXTDES
```

Figure 194. (Part 2 of 11)

```
S       SELECT  632IDP00/QUIK/SOURCE/DATA/VALPUT
$       LINK    DATACR,DATACH
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/CREATE
$       SELECT  632IDP00/QUIK/SOURCE/DATA/CREATE
$       SELECT  632IDP00/QUIK/SOURCE/DATA/VALPUT
$       LINK    DBMOD,DATA
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/DBMOD
$       SELECT  632IDP00/QUIK/SOURCE/DBMOD/DBMOD
$       SELECT  632IDP00/QUIK/SOURCE/DBMOD/DESTAB
$       LINK    INDXER,DBMOD
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/INDEXER
$       SELECT  632IDP00/QUIK/SOURCE/INDEXER/INDEXER
$       SELECT  632IDP00/QUIK/SOURCE/INDEXER/COMPLEX
$       SELECT  632IDP00/QUIK/SOURCE/INDEXER/CRTBLE
$       SELECT  632IDP00/QUIK/SOURCE/INDEXER/SETVAL
$       SELECT  632IDP00/QUIK/SOURCE/INDEXER/VLRADI
$       LINK    PLANS,INDEXER
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/PLANSET
$       SELECT  632IDP00/QUIK/SOURCE/PLANSET/PLANSET
$       SELECT  632IDP00/QUIK/SOURCE/PLANSET/ADJUSTGP
$       SELECT  632IDP00/QUIK/SOURCE/PLANSET/CALCOMP
$       SELECT  632IDP00/QUIK/SOURCE/PLANSET/GRPEM
$       SELECT  632IDP00/QUIK/SOURCE/PLANSET/PRINTGP
$       SELECT  632IDP00/QUIK/SOURCE/PLANSET/SRTTGT
$       SELECT  632IDP00/QUIK/SOURCE/PLANSET/TANKER
$       SELECT  632IDP00/QUIK/SOURCE/PLANSET/VLRADP
$       LINK    PREP,PLANS
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/PREPALOC
$       SELECT  632IDP00/QUIK/SOURCE/PREPALOC/PREPALOC
$       SELECT  632IDP00/QUIK/SOURCE/PREPALOC/CHGBAS
$       SELECT  632IDP00/QUIK/SOURCE/PREPALOC/FIXWEP
$       SELECT  632IDP00/QUIK/SOURCE/PREPALOC/GEOIN
$       SELECT  632IDP00/QUIK/SOURCE/PREPALOC/GEOPREP
$       SELECT  632IDP00/QUIK/SOURCE/PREPALOC/PRNPRP
$       SELECT  632IDP00/QUIK/SOURCE/PREPALOC/SETRD
$       SELECT  632IDP00/QUIK/SOURCE/PREPALOC/WEPIN
$       SELECT  632IDP00/QUIK/SOURCE/PREPALOC/WEPPREP
$       LINK    EDIT,PREP
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/EDITDB
$       SELECT  632IDP00/QUIK/SOURCE/EDITDB/EDITDB
$       LINK    ECOUNT
$       FORTY   MAP,XREF,DECK
```

Figure 194. (Part 3 of 11)

```
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/COUNTS
$       SELECT  632IDP00/QUIK/SOURCE/EDITDB/COUNTS
$       LINK    EGENED,ECOUNT
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/GENEDIT
$       SELECT  632IDP00/QUIK/SOURCE/EDITDB/GENEDIT
$       SELECT  632IDP00/QUIK/SOURCE/EDITDB/BUILDTAB
$       SELECT  632IDP00/QUIK/SOURCE/EDITDB/FORMLOC
$       SELECT  632IDP00/QUIK/SOURCE/EDITDB/SETFLD
$       SELECT  632IDP00/QUIK/SOURCE/EDITDB/SWITH
$       LINK    ENORMA,EGENED
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/NORMAL
$       SELECT  632IDP00/QUIK/SOURCE/EDITDB/NORMAL
$       LINK    EPROCE,ENORMA
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/PROCEDIT
$       SELECT  632IDP00/QUIK/SOURCE/EDITDB/PROCEDIT
$       SELECT  632IDP00/QUIK/SOURCE/EDITDB/XWITH
$       LINK    REPORT,EDIT
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/REPORT
$       SELECT  632IDP00/QUIK/SOURCE/REPORT/REPORT
$       SELECT  632IDP00/QUIK/SOURCE/REPORT/DSPPUT
$       SELECT  632IDP00/QUIK/SOURCE/REPORT/TABMNT
$       LINK    RPTDSN
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/DESIGN
$       SELECT  632IDP00/QUIK/SOURCE/REPORT/DESIGN
$       LINK    RPTALT,RPTDSN
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/ALTER
$       SELECT  632IDP00/QUIK/SOURCE/REPORT/ALTER
$       LINK    RPTDMK,RPTALT
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/DSPMAK
$       SELECT  632IDP00/QUIK/SOURCE/REPORT/DSPMAK
$       LINK    RPTPRN,RPTDMK
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/PRINT
$       SELECT  632IDP00/QUIK/SOURCE/REPORT/PRINT
$       SELECT  632IDP00/QUIK/SOURCE/REPORT/PRNATD
$       SELECT  632IDP00/QUIK/SOURCE/REPORT/XDEFN
$       LINK    SRM,REPORT
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/SRM
```

Figure 194. (Part 4 of 11)

```
$       SELECT  632IDP00/QUIK/SOURCE/SRM/SRM
$       LINK    EIM,SRM
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/EIM
$       SELECT  632IDP00/QUIK/SOURCE/EIM/EIM
$       SELECT  632IDP00/QUIK/SOURCE/EIM/CONVLL
$       LINK    BSIDAC
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/SIDAC
$       SELECT  632IDP00/QUIK/SOURCE/EIM/SIDAC
$       LINK    BOTHER,BSIDAC
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/BLDOTH
$       SELECT  632IDP00/QUIK/SOURCE/EIM/BLDOTH
$       SELECT  632IDP00/QUIK/SOURCE/EIM/XEDEFN
$       LINK    BTABLE,BOTHER
$       FORTY   MAP,XREF,DECK
$       PRMFL   632IDP00/QUIK/OBJECT/TABLE
$       SELECT  632IDP00/QUIK/SOURCE/EIM/TABLE
$       LINK    PLOTTT,BTABLE
$       FORTY   MAP,XREF,DECK
$       DRMFL   632IDP00/QUIK/OBJECT/PLOTDATA
$       SELECT  632IDP00/QUIK/SOURCE/EIM/PLOTDATA
$       SELECT  632IDP00/QUIK/SOURCE/EIM/PICS
$       SELECT  632IDP00/QUIK/SOURCE/EIM/PROJCT
$       LINK    PLOTIT,PLOTTT
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/PLOTIT
$       SELECT  631IDP00/QUIK/SOURCE/EIM/PLOTIT
$       SELECT  631IDP00/QUIK/SOURCE/EIM/FNDSRT
$       SELECT  631IDP00/QUIK/SOURCE/EIM/INTRPL
$       SELECT  631IDP00/QUIK/SOURCE/EIM/PICS
$       SELECT  631IDP00/QUIK/SOURCE/EIM/
$       SELECT  631IDP00/QUIK/SOURCE/EIM/PLOTINIT
$       SELECT  631IDP00/QUIK/SOURCE/EIM/PROJCT
$       SELECT  631IDP00/QUIK/SOURCE/EIM/SUBPLOT
$       SELECT  631IDP00/QUIK/SOURCE/EIM/SUBREAD
$       LINK    ALOC,EIM
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/ALOC
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/ALOC
$       LINK    ALCINT
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/ALCINT
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/INITAL
```

Figure 194. (Part 5 of 11)

```
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/STALL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/FORMATS
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/FMUP
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/LAMGET
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PREMIUMS
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PRNTOS
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/SALVAL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/SPLIT
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/WAD
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/WADOUT
$       LINK    DEFAL,STAL
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/DEFAL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/DEFALOC
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/FMUP
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/LAMGET
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PREMIUMS
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/PRNTOD
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/RESVAL
$       SELECT  632IDP00/QUIK/SOURCE/ALOC/SALVAL
$       LINK    EVAL,ALOC
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/EVALALOC
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/EVALALOC
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/EVALPLAN
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/EVAL2
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/PREVAL
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/SSSPCALC
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/TGTMODIF
$       SELECT  632IDP00/QUIK/SOURCE/EVALALOC/WPNMODIF
$       LINK    DGZSEL,EVAL
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/ALOCOUT
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/ALOCOUT
$       LINK    OFFSET
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/OFFSET
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/COMPRESS
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/CUMINV
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/DGZ
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/ERGOT1
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/FINDMIN
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/F2BMIN
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/GRADF
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/MOVE
```

Figure 194. (Part 7 of 11)

```
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/PERTBLD
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/PROCCOMP
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/SEECALC
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/SEEINPUT
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/VAL
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/VMARG
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/WEPGET
$       LINK    ASGSET,OFFSET
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/ASGSET
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/SUMPRN
$       LINK    MINIO,ASGSET
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/MINIO
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/MINIOUT
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/CONVLL
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/FINDTIME
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/INFORM
$       LINK    POST,DGZSEL
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/POSTALOC
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/POSTALOC
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/CENTROID
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/CHGPLAN
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/CORRPARM
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/DIFF
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/DUMPSRT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/EVALB
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/EVALOA
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/EVALOB
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/FLTPLAN
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/FLTROUTE
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/GENRAID
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/GETGROUP
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/GETSORT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/INITOPT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/INPOTGT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/NEXTFLT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/NOCORR
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/OPTRAID
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/OUTPOTGT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/OUTSRT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/PRERAID
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/PRINTIT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/PRNTF
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/SETFLAG
```

Figure 194. (Part 8 of 11) CH-2

```
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/PERTBLD
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/PROCCOMP
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/SEECALC
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/SEEINPUT
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/VAL
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/VMARG
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/WEPGET
$       LINK    ASGSET,OFFSET
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/ASGSET
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/SUMPRN
$       LINK    MINIO,ASGSET
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/MINIO
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/MINIOUT
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/CONVLL
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/FINDTIME
$       SELECT  632IDP00/QUIK/SOURCE/ALOCOUT/INFORM
$       LINK    POST,DGZSEL
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/POSTALOC
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/POSTALOC
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/CENTROID
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/CHGPLAN
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/CORRPARM
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/DIFF
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/DUMPSRT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/EVALB
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/EVALOA
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/EVALOB
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/FLTPLAN
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/FLTROUTE
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/GENRAID
$       SELECT  632IDPC0/QUIK/SOURCE/POSTALOC/GETGROUP
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/GETSORT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/INITOPT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/INPOTGT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/NEXTFLT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/NOCORR
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/OPTRAID
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/OUTPOTGT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/OUTSRT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/PRERAID
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/PRINTIT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/PRNTF
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/SETFLAG
```

Figure 194. (Part 8 of 11) CH-2

```
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/SORTOPT
$       SELECT  632IDP00/QUIK/SOURCE/POSTALOC/TGTASGN
$       LINK    FOOT,POST
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/FOOTPRNT
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/FOOTPRNT
$       LINK    OPTS
$       FORTY   MAP,XRED,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/FOOTOPTS
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/TAB2INPT
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/MKAOS
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/PRAOS
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/PRINSETS
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/TAOS
$       LINK    SETS,OPTS
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/FOOTSETS
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/NEWCOOR
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/SETDATA
$       LINK    PLAN,SETS
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/FOOTPLAN
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/AXES
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/DRIVER
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/ELLIPSE
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/PATHFIND
$       SELECT  632IDP00/QUIK/SOURCE/FOOTPRNT/XAOS
$       LINK    ASGN,PLAN
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/FOOTASGN
$       LINK    PLANO,FOOT
$       FORTY   MAP,XREF,DECK
$       PRMFL   X*,W,S,632IDP00/QUIK/OBJECT/PLANO
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/PLANOUT
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/CLINDATA
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/CONVLL
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/GEOGET
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/SNAPCON
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/SEPDATA
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/XLL
$       LINK    PLNT
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/PLNT
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/PLNTPLAN
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/ALTPLAN
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/ADJUST
```

Figure 194. (Part 9 of 11)

```
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/ALTERR
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/CHGTIM
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/DECOYADD
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/DISTIME
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/FINDME
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/FLTSORT
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/FLYPOINT
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/INITANK
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/KERPLUNK
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/LAUNCH
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/LNCHDATA
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/PLAN
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/PLANBOMB
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/PLANTMIS
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/POST
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/POSTLAUN
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/SNAPIT
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/SNAPOUT
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/SORBOMB
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/SWTCHALT
$       LINK    INTR,PLNT
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/INTR
$       SELECT  631IDP00/QUIK/SOURCE/PLANOUT/INTRFACE
$       SELECT  631IDP00/QUIK/SOURCE/PLANOUT/ABOUT
$       SELECT  631IDP00/QUIK/SOURCE/PLANOUT/FINDTIME
$       SELECT  631IDP00/QUIK/SOURCE/PLANOUT/IAZIM
$       SELECT  631IDP00/QUIK/SOURCE/PLANOUT/IFSET
$       SELECT  631IDP00/QUIK/SOURCE/PLANOUT/IFUNCT
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/INFORM
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/NOP
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/PRNTOFFS
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/RDCLAUSE
$       SELECT  632IDP00/OUIK/SOURCE/PLANOUT/STOUT
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/XSET
$       LINK    TANK,INTR
$       FORTY   MAP,XREF,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/TANK
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/PLANTANK
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/PRNTAB
$       SELECT  632IDP00/QUIK/SOURCE/PLANOUT/VAM
$       LINK    DMAKE,PLANO
$       FORTY   MAD,XRED,DECK
$       PRMFL   C*,W,S,632IDP00/QUIK/OBJECT/DATAMAKE
$       SELECT  632IDP00/QUIK/SOURCE/DATAMAKE/DATAMAKE
```

Figure 194. (Part 10 of 11)

```
$       SELECT  632IDP00/QUIK/SOURCE/DATAMAKE/CALCOMP
$       SELECT  632IDP00/QUIK/SOURCE/DATAMAKE/COMPLEX
$       SELECT  632IDP00/QUIK/SOURCE/DATAMAKE/CRTABLE
$       SELECT  632IDP00/QUIK/SOURCE/DATAMAKE/SRTTGT
$       SELECT  632IDP00/QUIK/SOURCE/DATAMAKE/VLRADI
$       SELECT  632IDP00/QUIK/SOURCE/DATAMAKE/VLRADP
$       EXECUTE DUMP,DEBUG,NREST,JREST
$       LIMITS  30,74K,-3K,30K
$       FFILE   P*,LGU/(06,42,43,11,12)
$       PRMFL   H*R/W,R,632IDP00/QUIK/COP/HSTAR
$       PRMFL   QD,R/W,R,632IDP00/QUIK/COP/IDS
$       PRMFL   UL,R/W,R,632IDP00/QUIK/LIBRARY/UTIL
$       PRMFL   PL,R,S,LIBRARY/PLOTLIB
$       FILE    02,X2R,100L
$       FILE    08,X8R,100L
$       FILE    19,X19R,100L
$       TAPE9   20,X20D,,12345,,INPUT-JAD
$       FILE    21,X21R,50L
$       FFILE   21,NBUFFS/2
$       FILE    22,X22R,50L
$       FFILE   22,NBUFFS/2
$       FILE    23,X23R,50L
$       FILE    23,NBUFFS/2
$       FILE    24,X24R,50L
$       FFILE   24,NBUFFS/2
$       FILE    25,X25R,100R
$       FILE    30,X30R,10L
$       TAPE9   31,X31D,,54321,,OUTPUT-SPILLTAPE
$       TAPE9   32,X32D,,54345,,SAVE-RESTORE-TAPE
$       TAPE9   35,X35D,,67890,,OUTPUT-TAPE-1
$       TAPE9   36,X36D,,98765,,OUTPUT-TAPE-2
$       DATA    I*
$       ENDJOB
***EOF
```

Figure 194. (Part 11 of 11)

THE CONTENTS OF THIS PAGE INTENTIONALLY LEFT BLANK

```
$       IDENT   1820510/30/5162,COMPL
$       USERID  632IDP99$PASSWORD/UZZ
$       LIMITS  30,60K,,10K
$       OPTION  FORTRAN,NOGO
$       LIBRARY UL,PL
$       LOWLOAD
$       ENTRY   ......
$       SELECT  632IDP00/QUIK/OBJECT/COP
$       SELECT  632IDP00/QUIK/OBJECT/QDATA
$       USE     .QMAX/1/,.QAREA/3126/,.QMIN/1/,.FRRD.
$       LINK    BOOTT
$       SELECT  632IDP00/QUIK/OBJECT/BOOT
$       LINK    TABSTR,BOOTT
$       USE     TABLZ/808/
$       LINK    ERRF
$       SELECT  632IPD00/QUIK/OBJECT/ERRFND
$       LINK    INPT,ERRF
$       SELECT  632IDP00/QUIK/OBJECT/INPTRN
$       LINK    JLM,TABSTR
$       SELECT  632IDP00/QUIK/OBJECT/JLM
$       LINK    ASSI
$       SELECT  632IDP00/QUIK/OBJECT/ASSIGN
$       LINK    SELE,ASSI
$       SELECT  632IDP00/QUIK/OBJECT/SELECT
$       LINK    ASTE,SELE
$       SELECT  632IDP00/QUIK/OBJECT/ASTERISK
$       LINK    DATA,JLM
$       SELECT  632IDP00/QUIK/OBJECT/DATA
$       LINK    DATADL
$       SELECT  632IDP00/QUIK/OBJECT/DELETE
$       LINK    DATACH,DATADL
$       SELECT  632IDP00/QUIK/OBJECT/CHANGE
$       LINK    DATACR,DATACH
$       SELECT  632IDP00/QUIK/OBJECT/CREATE
$       LINK    DBMOD,DATA
$       SELECT  632IDP00/QUIK/OBJECT/DBMOD
$       LINK    INDXER,DBMOD
$       SELECT  632IDP00/QUIK/OBJECT/INDEXER
$       LINK    PLANS,INDXER
$       SELECT  632IDP00/QUIK/OBJECT/PLANSET
$       LINK    PREP,PLANS
$       SELECT  632IDP00/QUIK/OBJECT/PREPALOC
$       LINK    EDIT,PREP
$       SELECT  632IDP00/QUIK/OBJECT/EDITDB
$       LINK    ECOUNT
$       SELECT  632IDP00/QUIK/OBJECT/COUNTS
```

Figure 195. H* Creation From Object (Part 1 of 4)

```
$       LINK    EGENED,ECOUNT
$       SELECT  632IDP00/QUIK/OBJECT/GENEDIT
$       LINK    ENORMA,EGENED
$       SELECT  632IDP00/QUIK/OBJECT/NORMAL
$       LINK    EPROCE,ENORMA
$       SELECT  632IDP00/QUIK/OBJECT/PROCEDIT
$       LINK    REPORT,EDIT
$       SELECT  632IDP00/QUIK/OBJECT/REPORT
$       LINK    RPTDSN
$       SELECT  632IDP00/QUIK/OBJECT/DESIGN
$       LINK    RPTALT,RPTDSN
$       SELECT  632IDP00/QUIK/OBJECT/ALTER
$       LINK    RPTDMK,RPTALT
$       SELECT  632IDP00/QUIK/OBJECT/DSPMAK
$       LINK    REPPRN,RPTDMK
$       SELECT  632IDP00/QUIK/OBJECT/PRINT
$       LINK    SRM,REPORT
$       SELECT  632IDP00/QUIK/OBJECT/SRM
$       LINK    EIM,SRM
$       SELECT  632IDP00/QUIK/OBJECT/EIM
$       LINK    BSIDAC
$       SELECT  632IDP00/QUIK/OBJECT/SIDAC
$       LINK    BOTHER,BSIDAC
$       SELECT  632IDP00/QUIK/OBJECT/BLDOTH
$       LINK    BTABLE,BOTHER
$       SELECT  632IDP00/QUIK/OBJECT/TABLE
$       LINK    PLOTTT,BTABLE
$       SELECT  632IDP00/QUIK/OBJECT/PLOTDATA
$       LINK    PLOTIT,PLOTTT
$       SELECT  632IDP00/QUIK/OBJECT/PLOTIT
$       LINK    ALOC,EIM
$       SELECT  632IDP00/QUIK/OBJECT/ALOC
$       LINK    ALCINT
$       SELECT  632IDP00/QUIK/OBJECT/ALCINT
$       LINK    ALCMUL,ALCINT
$       SELECT  632IDP00/QUIK/OBJECT/ALCMUL
$       LINK    FGD
$       SELECT  632IDP00/QUIK/OBJECT/FGD
$       LINK    SGD,FGD
$       SELECT  632IDP00/QUIK/OBJECT/SGD
$       LINK    STAL,SGD
$       SELECT  632IDP00/QUIK/OBJECT/STAL
$       LINK    DEFAL,STAL
$       SELECT  632IDP00/QUIK/OBJECT/DEFAL
$       LINK    EVAL,ALOC
$       SELECT  632IDP00/QUIK/OBJECT/EVALALOC
```

Figure 195. (Part 2 of 4)

```
$       LINK    DGZSEL,EVAL
$       SELECT  632IDP00/QUIK/OBJECT/ALOCOUT
$       LINK    OFFSET
$       SELECT  632IDP00/QUIK/OBJECT/OFFSET
$       LINK    ASGSET,OFFSET
$       SELECT  632IDP00/QUIK/OBJECT/ASGSET
$       LINK    MINIO,ASGSET
$       SELECT  632IDP00/QUIK/OBJECT/MINIO
$       LINK    POST,DGZSEL
$       SELECT  632IDP00/QUIK/OBJECT/POSTALOC
$       LINK    FOOT,POST
$       SELECT  632IDP00/QUIK/OBJECT/FOOTPRNT
$       LINK    OPTS
$       SELECT  632IDP00/QUIK/OBJECT/FOOTOPTS
$       LINK    SETS,OPTS
$       SELECT  632IDP00/QUIK/OBJECT/FOOTSETS
$       LINK    PLAN,SETS
$       SELECT  632IDP00/QUIK/OBJECT/FOOTPLAN
$       LINK    ASGN,PLAN
$       SELECT  632IDP00/QUIK/OBJECT/FOOTASGN
$       LINK    PLANO,FOOT
$       SELECT  632IDP00/QUIK/OBJECT/PLANO
$       LINK    PLNT
$       SELECT  632IDP00/QUIK/OBJECT/PLNT
$       LINK    INTR,PLNT
$       SELECT  632IDP00/QUIK/OBJECT/INTR
$       LINK    TANK,INTR
$       SELECT  632IDP00/QUIK/OBJECT/TANK
$       LINK    DMAKE,PLANO
$       SELECT  632IDP00/QUIK/OBJECT/DATAMAKE
$       EXECUTE DUMP,DEBUG,NREST,JREST
$       LIMITS  10,60K,-3K,10K
$       FFILE   PX,LGU/(06,42,43,11,12)
$       PRMFL   H*,R/W,R,632IDP00/QUIK/COP/HSTAR
$       PRMFL   QD,R/W,R,631IDP00/QUIK/COP/IDS
$       PRMFL   UL,R/W,R,631IDP00/QUIK/LIBRARY/UTIL
$       PRMFL   PL,R,S,LIBRARY/PLOTLIB
$       FILE    02,X2R,100L
$       FILE    08,X8R,100L
$       FILE    19,X19R,100L
$       TAPE9   20,X20D,,12345,,INPUT-JAD
$       FILE    21,X21R,50L
$       FFILE   21,NBUFFS/2
$       FILE    22,X22R,50L
$       FFILE   22,NBUFFS/2
$       FILE    23,X23R,50L
```

Figure 195. (Part 3 of 4)